AD-A083 792 ARMY MILITARY PERSONNEL CENTER ALEXANDRIA VA F/G 5/3

AN ANALYSIS OF THE MULTIPLE OBJECTIVE CAPITAL BUDGETING PROBLEM--ETC(U)

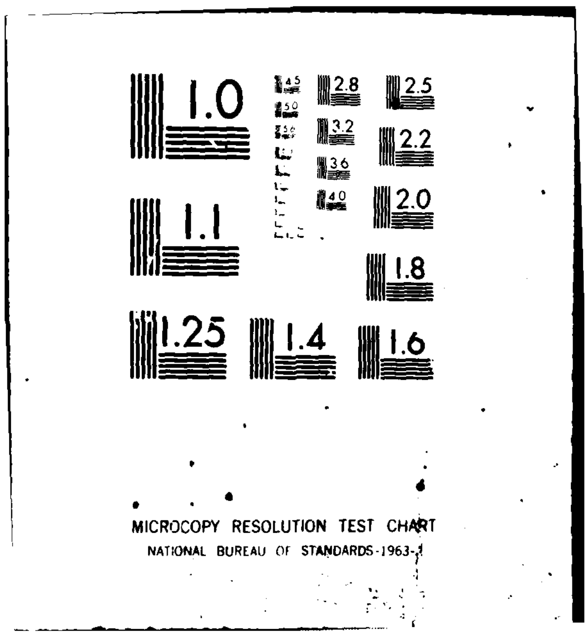
MAY 80 M G HEADLY

UNCLASSIFIED NL

1 OF 2

AD

A083792


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

LEVEL II

ADA083792

An Analysis of the Multiple Objective Capital Budgeting Problem Via Fuzzy Linear Integer (0-1) Programming

CPT. Michael G. Headly
HQDA, MILPERCEN (DAPC-OPP-E)
200 Stovall Street
Alexandria, VA 22332

May 31, 1980

A thesis submitted to The Pennsylvania State University, University Park, Pennsylvania, in partial fulfillment of the requirements for the degree of Master of Science in Industrial Engineering and Operations Research.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO. AD-A083 792

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)

(6) An Analysis of the Multiple Objective Capital Budgeting Problem via Fuzzy Linear Integer (0-1) Programming.

5. TYPE OF REPORT & PERIOD COVERED

(9) Final Report, 31 May 1980

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

(10) CPT. Michael G. Headly George

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Student, HQDA, MILPERCEN, ATTN: DAPC-OPP-E
200 Stovall Street, Alexandria, Va. 22332

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

HQDA, MILPERCEN, ATTN: DAPC-OPP-E
200 Stovall Street, Alexandria VA 22332

12. REPORT DATE

(11) 31 May 1980

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

(12) 154

15. SECURITY CLASS. (of this report)

Unclassifed

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Master's of Science Thesis Submitted to the Pennsylvania State University's Industrial Engineering and Operations Research Program. Portions of this thesis were presented at the TIMS/ORSA Joint National Meeting, Washington D.C., May 1980.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Linear Programming, Integer Programming, Capital Budgeting, Project and Program Selection, Heuristic Solution, Research and Development, Optimization, Mathematical Programming.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A multiple objective fuzzy linear programming approach to the capital budgeting problem is developed. Since much of the available data in any capital budgeting decision situation is either of an imprecise or ill-defined nature, a mathematical optimization technique is required that is capable of incorporating this inherent uncertainity. Fuzzy linear programming provides an effective methodology for this analysis.

Specifically, a mathematical model is developed which utilizes → next page

DD FORM 1473 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE

391191 AM

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. Continued

fuzzy linear programming as a solution technique for the research and development program or project selection problem. In addition an exchange heuristic,a modified form of C. C. Petersen's exchange algorithm, is presented.

A limited bibliography of works in multiple objective optimization is presented. Two computer codes are included.The first utilizes the IBM MPSX/Mixed Integer Programming procedures to solve the (0-1) linear integer programming problem. The second is a FORTRAN program to solve the exchange heuristic algorithm discussed previously.

The Pennsylvania State University

The Graduate School

An Analysis of the Multiple Objective Capital Budgeting Problem Via Fuzzy Linear Integer (0-1) Programming

A Thesis in

Industrial Engineering and Operations Research

by

Michael G. Headly

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

May 1980

We approve the thesis of Michael George Headly.

Date of Signature:

3/24/80 — James P. Ignizio

James P. Ignizio, Associate Professor of Industrial Engineering, Thesis Adviser

3/24/80 — William E Biles

William E. Biles, Professor of Industrial Engineering, Head of the Department of Industrial and Management Systems Engineering

3/24/80

Milton C. Hallberg, Professor of Agricultural Economics, Chairman of the Committee on Operations Research

3/24/80

Jack C. Hayya, Professor of Management Science

3/24/80

Emory E. Enscore, Associate Professor of Industrial Engineering

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

A multiple objective fuzzy linear programming approach to the capital budgeting problem is developed. Since much of the available data in any capital budgeting decision situation is either of an imprecise or ill-defined nature, a mathematical optimization technique is required that is capable of incorporating this inherent uncertainty. Fuzzy linear programming provides an effective methodology for this analysis.

Specifically, a mathematical model is developed which utilizes fuzzy linear programming as a solution technique for the research and development program or project selection problem. In addition, an exchange heuristic, a modified form of C. C. Petersen's exchange algorithm, is presented.

ACKNOWLEDGMENTS

The author expresses his gratitude to Dr. James Ignizio, his thesis adviser, for guidance during his work at The Pennsylvania State University.

The author is also grateful to Dr. Jack Hayya and Dr. Emory Enscore for their contribution to this thesis.

Support for the author's graduate work has been provided by the United States Army.

CHAPTER 1

INTRODUCTION

1.1 Purpose of the Research

The objective of the research documented in this thesis is the application and demonstration of a method for analysis of management decisions involving multiple objectives and constraints which are of a vague or ill-defined nature.

The traditional capital budgeting problem involves a single objective deterministic approach to the allocation of limited resources among available investment opportunities. The selection from among the various investment possibilities is such that the total return from the investment is maximized. In contrast to the traditional problem formulation, real-world capital investment decision analysis invariably encompasses nondeterministic systems involving multiple and usually conflicting objectives.

Investment selection or program selection in research and development planning is a multifaceted decision regularly faced by decision makers in government, industry, and the military. The constantly expanding nature of technological development necessitates decisions that involve multiple objectives in the decision criteria. Simply maximizing total return is an unrealistic and oversimplified decision criterion.

The complex selection process of research development programs may include the consideration of numerous factors, some of which are monetary while others are nonmonetary in nature. Influencing factors,

whose primary concern is not income generating, are demonstrated in safety and environmental considerations, which are inherent in virtually all business decisions today. The decision maker is clearly faced with a decision situation which is characterized mathematically as multiple criteria decision making.

Many mathematical programming techniques have been employed as a means of solving the capital budgeting problem; and, specifically, the investment or program selection problem has received a great deal of attention. A relatively new multiple objective optimization technique is fuzzy linear programming.

Fuzzy linear programming with its foundation in the theory of fuzzy sets is an optimization methodology designed for problems that are either too vague or too ill-defined to allow analysis by classical mathematical techniques. The inherent uncertainty which is ever present in any capital investment decision is the motivating influence in an examination of the applicability of fuzzy linear programming as a solution technique for the capital budgeting problem.

The design of this study encompasses five main objectives. These are:

1. Review various mathematical programming methodologies so as to establish applicability to the capital budgeting problem.
2. Evaluate the applicability of fuzzy linear programming as a solution technique for the capital budgeting problem.

3. Develop a fuzzy linear integer programming algorithm to solve the capital budgeting problem.
4. Apply the fuzzy linear integer programming algorithm to a representative problem.
5. Discuss extensions of this study and identify additional areas to which fuzzy programming techniques have applicability.

1.2 Organization of the Paper

This paper is organized as follows. Chapter 2 includes a historical perspective of various methodologies that have been employed as solution techniques for the capital budgeting problem. In Chapter 3, the basic elements of the theory of fuzzy sets are reviewed. Decision making in a fuzzy environment is discussed, and the model of fuzzy linear programming is presented in Chapter 4. In Chapter 5, the fuzzy capital budgeting model is presented along with the solution algorithm. Two example problems are solved. The results of the study are reviewed in Chapter 6, as well as possible extensions, and additional areas of applicability are suggested.

CHAPTER 2

HISTORICAL PERSPECTIVE OF THE CAPITAL BUDGETING PROBLEM

2.1 General

Decision makers have always sought a means of analyzing alternative investment possibilities in an efficient manner. The past twenty-five years have seen the development of analytical techniques to provide this analysis. The development of numerous quantitative analysis techniques has provided decision makers with a framework to more efficiently conduct this analysis. The usefulness of these quantitative techniques has been greatly extended with the ever-increasing accessibility of computers. While the computer's capability to analyze and store data has increased tremendously, the cost has steadily decreased. Today, the use of computer technology is widespread. Since the cost of many computers is no longer prohibitive, many small industries are utilizing quantitative analysis techniques that previously were reserved for government and large industries.

The classical approach to the analysis of alternate investment possibilities has been the maximization or minimization of a single objective function. Traditionally, this objective has been the maximization of profits or the minimization of costs. A significant amount of discussion has been generated concerning the classical approach and its inapplicability to today's complex decisions [1-11]. The basis of single objective function mathematical modelling is lost when it is recognized that real decision makers do not attempt to

optimize a single objective function. Rather, a solution is sought that satisfies the numerous objective functions that characterize a decision process. The solution is a compromise from among the various objective functions [5, 10, 12, 13]. The compromise is the result of the real-world limitations imposed on decision makers.

2.2 Survey of Related Literature

The multiple objective function optimization technique of fuzzy linear programming is a relatively new approach to multiple criteria decision making. Zimmermann [9, 10, 11] has shown the mathematical feasibility of this approach and its application to the media selection problem originally posed by Charnes et al. [14]. Two extensive bibliographies have been published on works related to fuzzy systems [15, 16]. A search of the literature failed to identify additional works dealing with the application of fuzzy linear programming as a multiple objective optimization technique. Kickert [17] has recently published a work detailing the various fuzzy theories and their impact on decision-making processes. Yager [18] discusses an eigenvector approach to the multiple objective optimization problem using fuzzy sets. There is increasing interest in multiple criteria decision making; and, correspondingly, a great deal of literature is available related to this work. The following paragraphs summarize a survey of the current literature on multiple criteria decision making, with an emphasis toward the capital budgeting problem.

A conference proceedings including numerous works on multiple criteria decision making was published by the University of South Carolina. A bibliography on multiple criteria decision making is included [19].

One mathematical programming technique that has been utilized for years as an optimization technique is linear programming. Charnes and Cooper [20] demonstrated an early use of linear programming as a solution technique for the problem of allocating funds. In recent years, multiple objective linear programming techniques have been developed. Benayoun, Larichev, de Montogolfier, and Tergny [21] discuss a methodology of using linear programming with multiple objectives. Belenson and Kapur [22] present an algorithm for solving multi-criteria linear programming problems with several examples. A multi-objective linear programming methodology has been presented by Evans and Steuer [1].

Goal programming is another robust optimization technique for dealing with decision problems involving multiple objectives. This technique was developed by Charnes and Cooper in the early 1950's [23]. Goal programming is an effective modelling methodology which affords an analysis of problems involving multiple, and possibly, conflicting objectives. The methodology requires an assignment of a priority to each objective. This priority assignment is a preemptive prioritization of the objectives in accordance with the priorities of the decision maker. Lee [13] published the first book entirely devoted to linear goal programming. Ijiri [24] in his work developed the concept of preemptive prioritization of objectives. Numerous applications of goal programming are available. These include capital budgeting optimization [4, 5, 8, 25, 26]; manpower planning [27]; academic planning, financial planning, and economic planning [13]; antenna array design and transportation problem [5]; and media

planning [14]. Survey works of goal programming have been published by Kornbluth [28] and Ignizio [3].

Integer and nonlinear goal programming algorithms have been developed and have realized many successful applications [5, 29]. Research is continuing to extend goal programming into the area of stochastic analysis. Contini [30] has demonstrated the mathematical feasibility of such an approach.

Interactive programming is yet another multi-criteria programming approach currently being utilized. The decision maker in this approach is required to specify trade-offs between the various objective functions. The process of specifying trade-offs is continued in a successive manner until no further trade-offs are desired by the decision maker. Geoffrion, Dyer, and Feinberg [31] demonstrate the application of interactive programming, while Zionts and Wallenius [32] present an overview of the interactive programming method as applied to the multiple criteria problem. Dyer [33] has also proposed an interactive goal programming technique, while Steuer [34, 35] has proposed an interactive approach to multiple objective linear programming.

Numerous other mathematical programming techniques have been discussed as solution methods for the multiple criteria decision problem. One technique that has received a great deal of attention is integer programming. The literature has many examples of the successful application of integer programming. Seward, Plane, and Hendrick [36] present an application in the area of allocating municipal funds for fire protection, Armstrong and Willis [37] discuss its use in the selection of water projects in California, and Nackel,

Goldman, and Fairman [38] demonstrate the use of integer programming in an example in the health care field. Chiu and Gear [39] present a stochastic integer programming approach to the research and development project selection problem.

A few of the other mathematical programming techniques with applications in the multiple criteria decision-making area are branch and bound procedures, dynamic programming and heuristic programming. Shih [40] has written on a branch and bound method, Kepler and Blackman [41] have demonstrated the use of dynamic programming in the selection of research and development projects, and Petersen [42, 43] has developed heuristic algorithms using exchange operations to solve the capital budgeting problem.

The recognition of the inherent risk and uncertainty in capital budgeting problems has been presented in many works in the literature. Hillier [44] presents a basic model for capital budgeting of risky interrelated projects. Stochastic analysis was initially proposed by Charnes and Cooper [45]. Their technique was termed chance-constrained programming. Healy [46] and Armstrong and Balintfy [47] have presented chance-constrained programming algorithms. Odom and Shannon [48] and Park and Theusen [49] have recently published works aimed at risk resolution in the capital budgeting decision analysis. Utility theory has also been a frequently employed technique in multicriteria decision making. Recent works in the literature include: Crawford, Huntzinger, and Kirkwood's [50] use of multiattribute utility theory in the selection of components of an electrical transmission system, and Keefer's [51] multiobjective analysis of research and development projects through the use of a multiattribute utility function.

The increased use of multiple criteria decision analysis is evident in the literature. Many excellent overviews of multiple objective optimization techniques are available. MacCrimmon [52] has analyzed the various techniques that are not mathematical programming approaches. These approaches involve either weighting factors methods, sequential elimination methods, or spatial proximity methods. Easton [2] reviews a variety of multivalued alternative weighting methods. Ignizio [3] reviews goal programming as a multiple objective optimization technique. Plane [53] presents integer programming and network analysis techniques, and Hax [54] discusses the use of decision analysis. Two survey papers [55, 56] discuss the use of the various decision-making techniques as related specifically to the capital budgeting problem.

The development of new approaches to the multiple criteria decision problem and the variety of applications of the more established techniques indicate a tremendous interest in multiple criteria optimization methodologies.

CHAPTER 3

BASIC FUZZY SET THEORY

3.1 The Decision-Making Process

The analysis of alternative courses of action culminating in a decision is an extremely complex process for the human mind. The complexity of real-world decision problems far exceed the capacity of the human mind to formulate and subsequently arrive at a reasonable solution [12]. The essence of a decision is that the decision maker is able to exercise his prerogative. Obviously, then, the decision maker must be faced with a situation involving several alternatives about which information is available. This information may be of a precise or exact type, or it may be vague or ill-defined. An effective decision-making process is normally an iterative process with a feedback capability so that, at various stages, additional information may enter into the analysis. The decision-making process with feedback is shown in Figure 1.

Decision making utilizing the multiple objective optimization technique of fuzzy linear programming is an effective methodology in which to employ this feedback process. Prior to any elaboration on fuzzy linear programming, a brief discussion of the basic principles of fuzzy set theory is necessary.

3.2 Fuzzy Set Theory

The theory of fuzzy sets was developed in response to a need for a conceptual framework to deal with problems which were either too

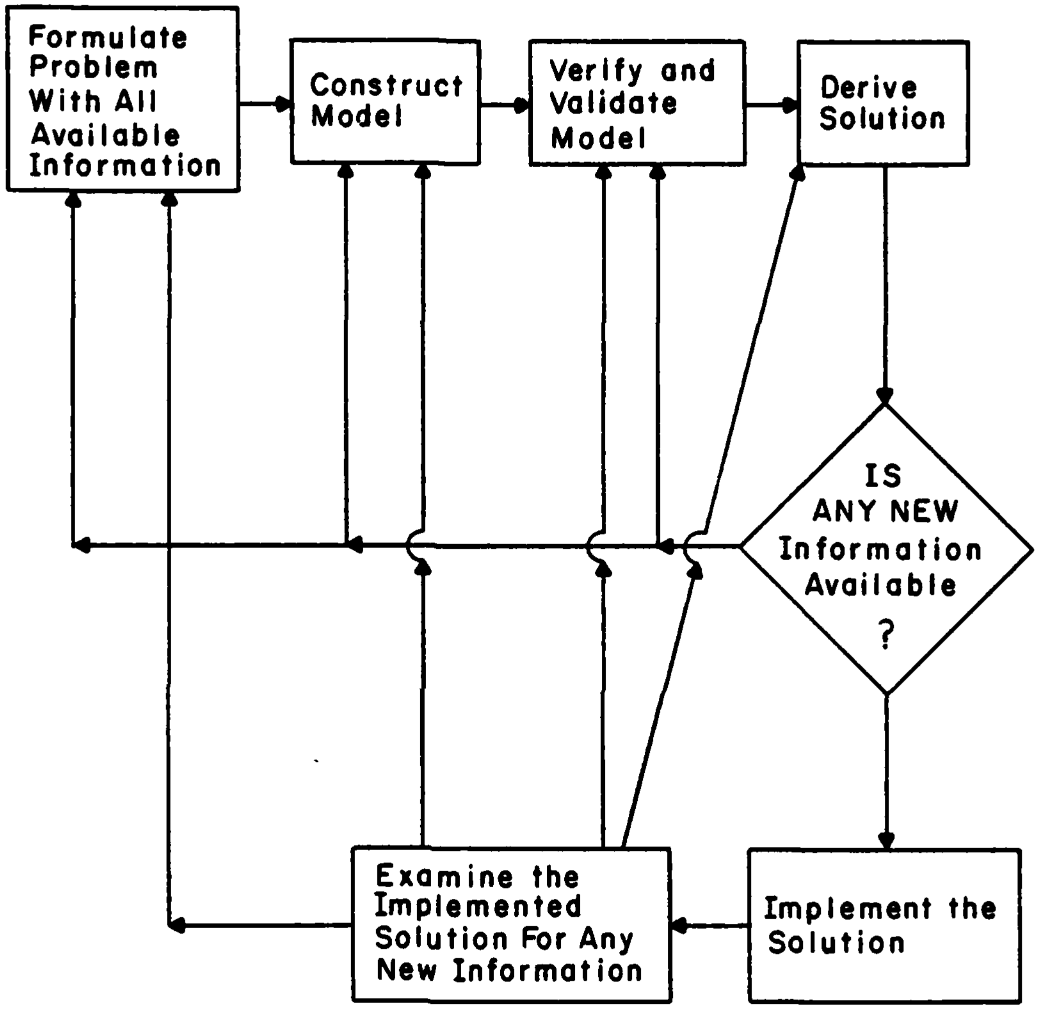


Figure 1. Decision Making as a Feedback Process

complex or too ill-defined to allow analysis by classical mathematical techniques.

Classical mathematics are much too rigid to be utilized in the optimization of systems that are humanistic in nature. These systems are composed largely of human perceptions and human judgments. Such systems are those in the fields of economics, psychology, sociology, linguistics, management science, medicine, law, philosophy, and others whose basic tenents are imprecise or fuzzy in nature.

The theory of fuzzy sets is founded on the theory of classes. Events may be viewed as in a continuum with respect to their membership or nonmembership in a class. The degree of membership in a class is the fundamental concept in the theory.

Classical mathematics' precise formulation of decision situations does not allow for the inclusion of a decision maker's judgmental capability. The concepts of fuzzy set theory create an overlap of the decision maker's judgmental ability and his quantitative analysis capabilities. The judgmental capability of the human mind analyzes a situation in an imprecise or approximate manner.

This imprecise or approximate analysis is necessitated by the complexity of today's managerial decision requirements. Real-life problems present themselves daily in vague or ill-defined ways. Many phenomena exist such as "satisfactory profits," "adequate return on investment," or "better productivity." None of these problems could be defined in precise mathematical terms. Instead, they would be twisted so as to conform to a precise mathematical optimization technique; and, therefore, the derived solution may or may not be accurate. In our attempts to understand and optimize systems which

are composed of various humanistic subsystems, the solutions obtained may pretend a higher degree of preciseness than is actually possible to achieve in the real system [57].

Fuzzy set theory provides a formal mathematical theory to analyze systems that are vague or inexact, with the vague or inexact nature defined by a fuzzy set [58].

3.3 Basic Definitions of the Theory of Fuzzy Sets

Zadeh [57] introduced the theory of fuzzy sets through the theory of sets, a generally universal mathematical theory. A set is defined as consisting of a finite or infinite number of elements [59]. The characteristic function of a set enables us to discuss the membership of the set in terms of functions. To define the characteristic function of a set, let A be a subset of the universe [60]. The function χ_A, the characteristic function, can only take on the values 0 or 1. If the universe is $X = \{x\}$, then χ_A is defined by the following:

$$\chi_A(x) = 1 \quad \text{if} \quad x \in A \ ,$$
$$\chi_A(x) = 0 \quad \text{if} \quad x \notin A \ .$$

Zadeh [57] utilized this concept of the characteristic function in his development of fuzzy set theory. Instead of the characteristic function being limited to only taking on the values 0 or 1, it is generalized to assume an infinite number of values between 0 and 1.

The basic definitions of fuzzy set theory which are important in the development of fuzzy linear programming will be presented in the following pages. These definitions are summarized from presentations by Zimmermann [9, 10, 11, 16] and Kickert [17].

<u>Fuzzy Set</u> - A class with a continuum of grades of membership.

Let X be a space of points (objects), with a generic element of X denoted by x_1, then, $X = \{x\}$. The fuzzy set A in X is characterized by a membership function $\mu_A(x)$ which associated with each point in X a nonnegative real number whose supremum is finite, with $\mu_A(x)$ representing the grade of membership of A in X. This is represented as:

$$A = \{x, \mu_A(x) | x \varepsilon X\} \quad ,$$

where $\mu_A(x)$ is the membership function of A in X.

<u>Example</u>: In the field of psychology, and specifically related to learning theory, the concepts of performance, learning, motivation, and anxiety are critical in the prediction of the outcome of any learning acquisition task. Let

$$X = \{0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$$

be possible scores which an individual may attain on a learning acquisition task. Fuzzy set A, "Motivation Levels Affecting Learning Acquisition," may be defined for a certain individual as:

$$A = \{(10, 0.2), (20, 0.4), (30, 0.6), (40, 0.65), (50, 0.7), (60, 0.75), (70, 0.85), (80, 1.0), (90, 0.9), (100, 0.8)\} \quad .$$

Fuzzy set B, "Anxiety Levels Affecting Learning Acquisition," may be stated in a similar manner for the same individual as follows:

$$B = \{(10, 0.1), (20, 0.3), (30, 0.5), (40, 0.60), (50, 0.65), (60, 0.75), (70, 0.85), (80, 0.95), (90, 1.0), (100, 0.85)\} \quad .$$

Graphically, these two fuzzy sets are shown in Figure 2.

<u>Intersection</u> - In set theory, the intersection of two sets A and B, written $A \cap B$, is the set C containing all elements common to A and B. In fuzzy set theory, the membership function of $A \cap B$ is defined as:

$$\mu(x) = \text{Min } [\mu_A(x), \mu_B(x)] \quad \text{for all } x \epsilon X \quad .$$

<u>Example</u>: In the learning theory example, the fuzzy set representing the intersection of fuzzy sets A and B would be the fuzzy set C. Fuzzy set C is defined as:

$$C = \{(10, 0.1), (20, 0.3), (30, 0.5), (40, 0.60), (50, 0.65), (60, 0.75), (70, 0.85), (80, 0.95), (90, 0.90), (100, 0.80)\} \quad .$$

<u>Union</u> - In set theory, the union of two sets A and B, written $A \cup B$, is the set D containing all elements in either A or B, or both. In fuzzy set theory, the membership function of $A \cup B$ is defined as:

$$\mu(x) = \text{Max } [\mu_A(x), \mu_B(x)] \quad \text{for all } x \epsilon X \quad .$$

<u>Example</u>: In the learning theory example, the fuzzy set representing the union of fuzzy sets A and B would be the fuzzy set D. Fuzzy set D is defined as:

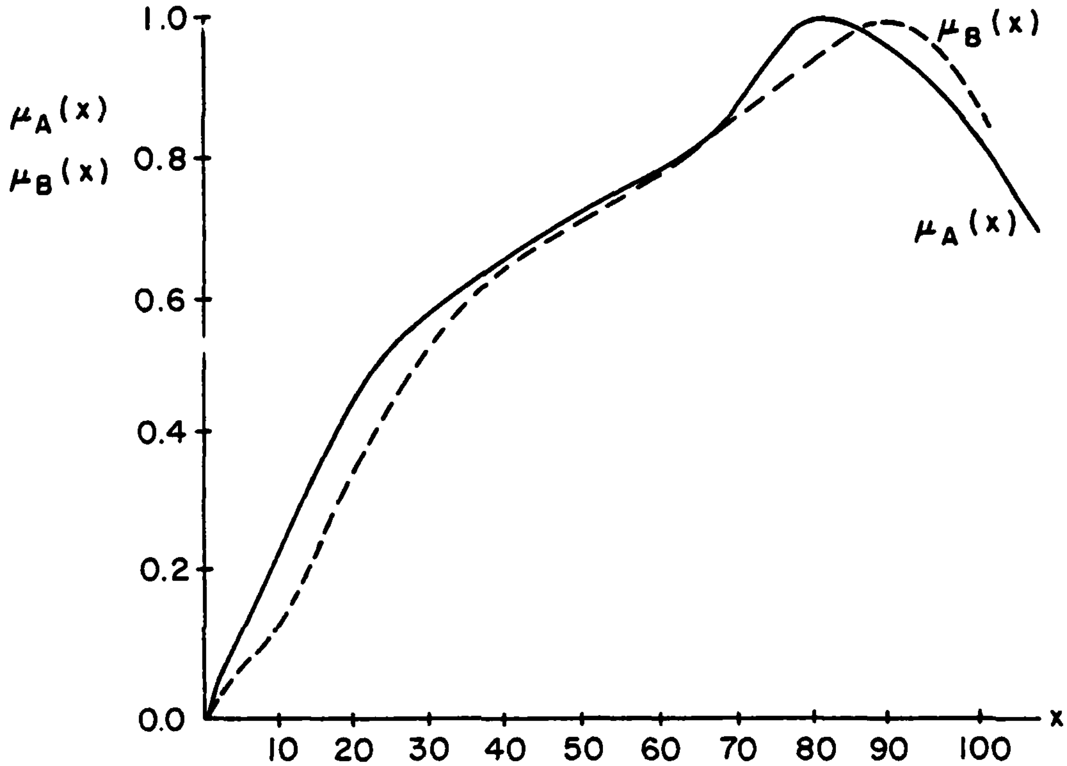


Figure 2. Illustration of Fuzzy Sets A and B

$$D = \{(10, 0.2), (20, 0.4), (30, 0.6), (40, 0.65), (50, 0.7), (60, 0.75), (70, 0.85), (80, 1.0), (90, 1.0), (100, 0.85)\} .$$

The union of fuzzy sets A and B is displayed in Figure 3, and the intersection of the two fuzzy sets is shown in Figure 4.

Equality - Two fuzzy sets are equal if

$$\mu_A(x) = \mu_B(x) \quad \text{for all } x \varepsilon X .$$

Normality - The definition of the membership function did not limit the values $\mu(x)$ could assume. If the supremum of the membership function equals 1, then the fuzzy set is called normal. This is defined as:

$$\text{Sup}_X \mu_A(x) = 1 .$$

A fuzzy set can be normalized by dividing $\mu_A(x)$ by $\text{Sup}_X\ \mu_1(x)$.

Algebraic Product - The algebraic product of two fuzzy sets A and B is denoted AB and is defined in terms of the membership functions of the fuzzy sets A and B .

$$\mu_{AB}(x) = \mu_A(x) \cdot \mu_B(x) .$$

Algebraic Sum - The algebraic sum of two fuzzy sets A and B is denoted by (A + B) and is defined in terms of the membership functions of the fuzzy sets A and B .

$$\mu_{(A+B)}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x) .$$

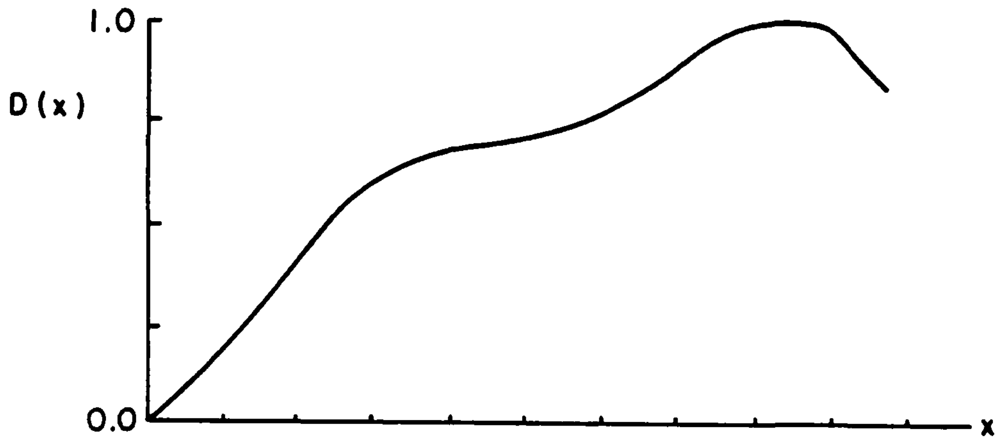


Figure 3. Union of Fuzzy Sets A and B

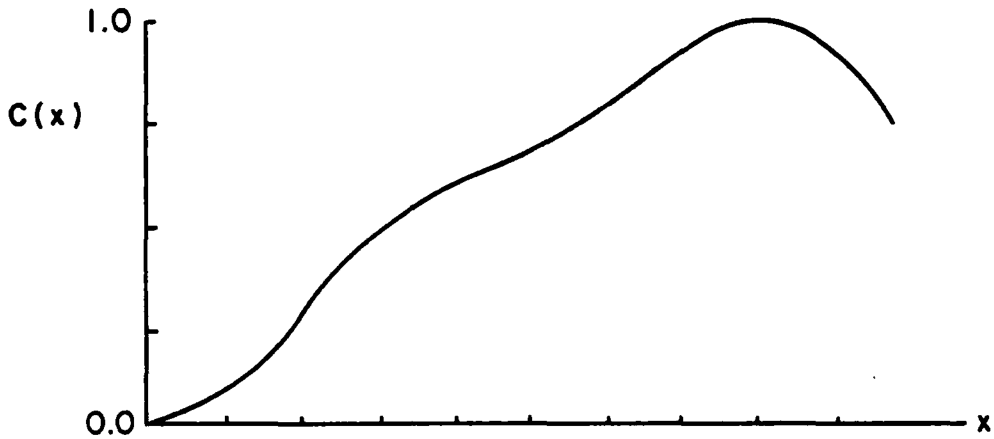


Figure 4. Intersection of Fuzzy Sets A and B

<u>Containment</u> - The fuzzy set definition of containment is analogous to the set theory definition of a subset. Fuzzy set A' is contained in fuzzy set B' if the membership function of A' is less than or equal to that of B' everywhere on X .

The basic definitions presented are sufficient for the discussion of fuzzy linear integer programming; however, there are many more concepts in the overall theory of fuzzy sets. For a more extensive treatment of the theory of fuzzy sets, Kaufmann [61] presents a complete review of the general theory of fuzzy sets.

CHAPTER 4

DECISION MAKING IN A FUZZY ENVIRONMENT

4.1 Fuzzy Decisions

In traditional decision making, the optimal decision is the selection of the activity or program with the highest desirability. In fuzzy decision making, the objective function(s) as well as the constraints may be fuzzy sets, each characterized by their membership functions. The optimal decision in the fuzzy environment is the fuzzy set formed by the intersections of the fuzzy sets describing the objective function(s) and constraints. Figure 5 illustrates the fuzzy decision process.

The region of intersection is a fuzzy set representing those activities which simultaneously satisfy the objective function(s) and the constraints. A solution to this fuzzy situation would be to select that point in the region of intersection with the greatest desirability or the highest degree of membership in the fuzzy set formed in the fuzzy decision. The selection of this solution point is analogous to the geometric representation of a solution to a linear programming problem [62]. The determination of the solution to the linear programming problem involving the intersection of n fuzzy sets is one of the basic principles in the development of fuzzy linear integer programming.

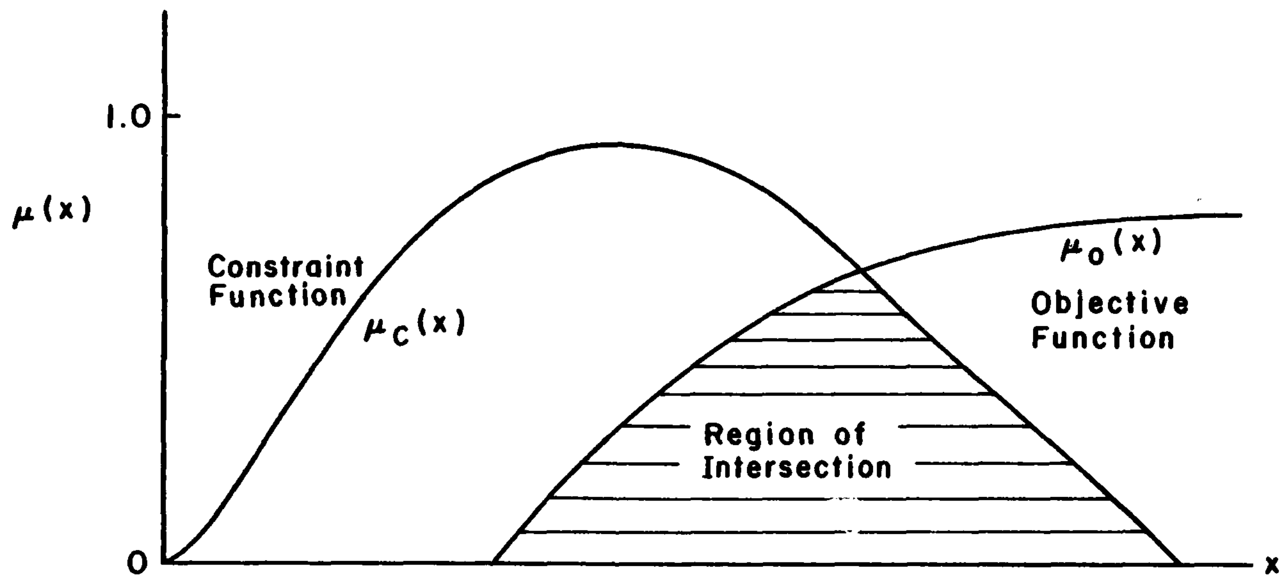


Figure 5. Fuzzy Decision Process

4.2 Fuzzy Linear Programming

The extension of fuzzy set theory into linear programming was utilized by Zimmermann [9]. The development of the fuzzy linear programming problem is as follows:

Start with the traditional vector minimization problem.

$$\begin{aligned} \textit{Minimize} \quad & \bar{Z} = \bar{C}\,\bar{x} \\ \text{Subject to} \quad & A\bar{x} \leq \bar{b} \\ & \bar{x} \geq 0 \quad . \end{aligned}$$

The fuzzy version of this same linear programming problem is:

$$\begin{aligned} \bar{C}\,\bar{z} & \lesssim \bar{Z}^{\circ} \\ A\,\bar{x} & \lesssim \bar{b} \\ \bar{x} & \geq 0 \quad , \end{aligned}$$

where

$\bar{C}$ is the vector of coefficients of the objective functions,

$\bar{b}$ is the vector of constraints,

A is the coefficient matrix, and

$\bar{Z}^{\circ}$ is the vector of aspiration levels of the fuzzy objectives and constraints.

The membership function $\mu(x)$ is defined such that it complies with the definition of a fuzzy set [57], that is, a real number in the interval (0,1).

$$\mu(x) = \begin{cases} 1 \text{ if } A\bar{x} \leq \bar{b} \text{ and } \bar{C}\,\bar{x} \leq \bar{Z} \text{ is satisfied} \\ 0 \text{ if } A\bar{x} \leq \bar{b} \text{ and } \bar{C}\,\bar{x} \leq \bar{Z} \text{ is strongly violated.} \end{cases}$$

The concept of an objective function being strongly violated or weakly violated is an important aspect of the decision-making process in a fuzzy environment. The membership function in Figure 6 will be utilized

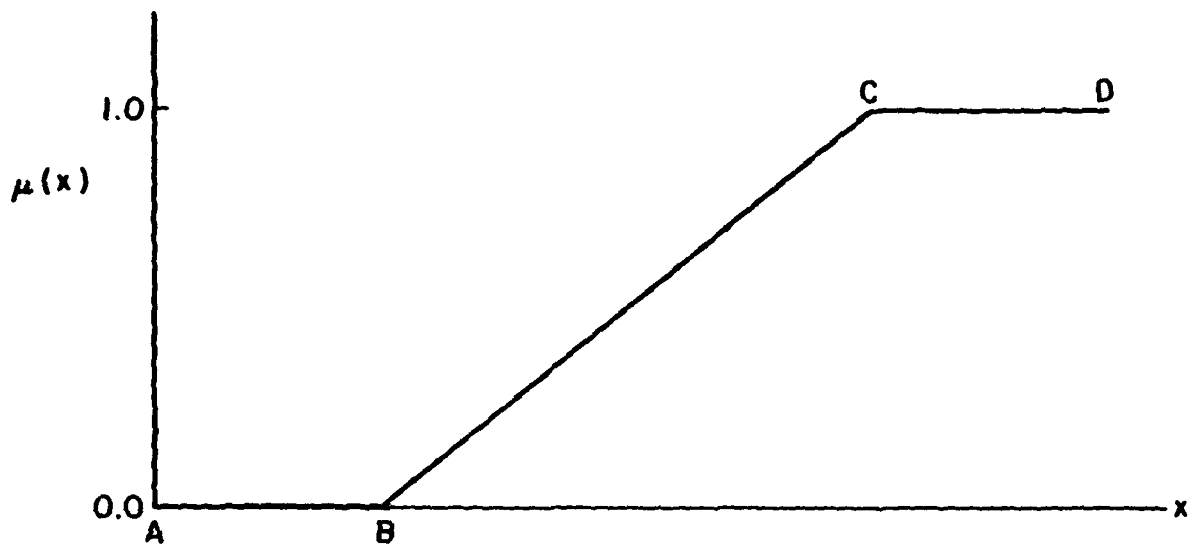


Figure 6. Membership Function of Fuzzy Set A

to illustrate this principle. Let this membership function be referred to as $\mu(x)$. In the interval CD , the membership function $\mu(x)$ is completely satisfied. The function describing the fuzzy set in this interval either achieves the aspiration level or exceeds it. In the interval BC , the membership function $\mu(x)$ is weakly violated. In this interval, the aspiration level is not achieved; however, the functional evaluation is greater than the lowest admissible value (Point B). The decision in this interval lies within the range of acceptable solutions as specified by the decision maker. In the interval AB , the membership function $\mu(x)$ is strongly violated. In this interval, any decision would lie wholly outside the acceptable range of solutions, since the functional evaluation of the fuzzy set would be less than the lowest admissible value, as specified by the decision maker.

If we let the fuzzy set B represent the intersection of the fuzzy sets representing the objective functions and the constraints, then the membership function of fuzzy set B is:

$$\mu_B(x) = \mu_o \cap \mu_C \quad .$$

The intersection of these two fuzzy sets is defined by the min operator to be:

$$\mu(Bx) = \underset{i}{\text{Min}}\ \mu_i \ ; \quad x \geq 0 \quad .$$

The maximizing decision is simply

$$\underset{x>0}{\text{Max}}\ \underset{i}{\text{Min}}\ [\mu_i(Bx)_i] \quad ,$$

which minimizes the maximum violation of the membership function.

If the solution technique is to be linear programming, the following assumptions are necessary [11]:

1. All objective functions must have a specified aspiration level. The objective functions are expressed in the form

$$C_i\overline{x} \lesssim Z_i \quad , \qquad i = 1, 2, \ldots, n \quad .$$

2. If the objective functions are in the same form as the constraints, then the problem may be formulated in the following form:

$$A\overline{x} \lesssim \overline{b} \qquad ,$$

where

A is the matrix of coefficients, and

$\overline{b}$ is the vector of aspiration levels of the objectives and the right-hand side values of the constraints.

3. The functions are assumed to be linear over the interval of consideration.

Given that assumption number (3) is satisfied, the linear membership function of fuzzy set B , the solution set of the intersection of the fuzzy sets representing the objectives and the constraints is:

$$\mu_B(x)_i = \begin{cases} 1 & \text{if} \quad (Bx)_i \leq b_i' \\ 1 - \dfrac{(Bx)_i - b_i'}{d_i} & \text{if} \quad b_i' < (Bx)_i \leq b_i' + d_i \\ 0 & \text{if} \quad (Bx)_i > b_i' + d_i \end{cases} \qquad ,$$

where

i indicates the i^{th} row of matrix B or b' ,

B is A , the coefficient matrix, augmented by the rows of the objective functions,

b' is the vector of the right-hand side values augmented by the upper bounds of the objective functions, and

d_i is the subjectively selected value of admissible violation.

By substituting

$$b_i'' = \frac{b_i'}{d_i} \quad \text{and} \quad B_i' = \frac{B_i}{d_i}$$

into the function $\mu_B(x)$, the maximizing decision then becomes:

$$\max_{x \geq 0} \ \min_{i} \ [b_i'' - (B'x)_i]$$

or

$$\max_{z \geq 0} \ \mu_D(x) \quad ,$$

where $\mu_D(x)$ represents the membership function of the fuzzy set representing the decision set.

It has been shown that the solution to this problem is equivalent to the following linear programming problem [9, 10, 11]:

$$\begin{aligned} &\text{Maximize} && \lambda \\ &\text{Subject to} && \lambda \leq b_i'' - (B'x)_i \quad , \quad i = 0, 1, \ldots, n \\ & && x > 0 \quad . \end{aligned}$$

To demonstrate a continuous fuzzy linear programming problem, consider the following example:

$$\text{Maximize} \quad Z = 4x_1 + 6x_2 + 8x_3 + 10x_4$$

$$\text{Subject to} \quad x_1 + 3x_2 + 4x_3 + 2x_4 \leq 40$$

$$3x_1 + 2x_2 + 3x_3 + 6x_4 \leq 60$$

$$4x_1 + x_2 + 2x_3 + 3x_4 \leq 50 \quad .$$

Solving this linear programming problem with the IBM MPSX mathematical programming system, the resulting program :

$$\bar{x} = (0,\ 8.57,\ 0,\ 7.14)$$

and

$$Z = 122.86 \quad .$$

The problem when formulated into the fuzzy linear programming equivalent utilizing the subjectively selected d_i values follows. The aspiration levels and the lowest admissible values as well as the allowable admissible ranges are shown in Table IV.1.

Table IV.1. Selected Values for Fuzzy Transformations

	$\mu = 0$	$\mu = 1$	d_i
Objective function	115	140	25
First constraint	50	40	10

where

$\mu = 0$ decision maker specified lowest admissible value,

$\mu = 1$ decision maker specified aspiration level,

d_i decision maker specified range of acceptable values.

The resulting fuzzy linear programming formulation is:

$$\text{Maximize} \quad \lambda$$

$$\begin{aligned}
\text{Subject to} \quad & \lambda \leq -4.6 + 0.16x_1 + 0.24x_2 + 0.32x_3 + 0.4x_4 \\
& \lambda \leq 5 - 0.1x_1 - 0.3x_2 - 0.4x_3 - 0.2x_4 \\
& 3x_1 + 2x_2 + 3x_3 + 6x_4 \leq 60 \\
& 4x_1 + x_2 + 2x_3 + 3x_4 \leq 50 \quad .
\end{aligned}$$

The solution to the fuzzy linear programming formulation is compared to the linear programming solution in Table IV.2. The fuzzy linear programming problem was solved using the IBM MPSX mathematical programming system.

Table IV.2. Summary of Calculations

Linear Programming	Fuzzy Linear Programming
$x_1 = 0.0$	$x_1 = 0.0$
$x_2 = 8.57$	$x_2 \quad 10.59$
$x_3 = 0.0$	$x_3 = 0.0$
$x_4 = 7.14$	$x_4 = 6.47$
$Z = 122.86$	$Z = 128.24$

The first advantage of fuzzy programming is that the decision maker is not required to specify in a precise manner the parameters of a decision situation. The decision maker is able to specify ranges of acceptability for those objective and constraint functions represented by fuzzy sets. In this example problem, the flexibility obtained in

the use of fuzzy linear programming enabled the decision maker to realize a greater return.

The second advantage of fuzzy programming is the ease with which it can be converted into a conventional mathematical programming problem. This is important due to the current availability of many mathematical programming techniques and algorithms [17].

CHAPTER 5

ZERO-ONE CAPITAL BUDGETING ALGORITHM

5.1 The Capital Budgeting Problem

The traditional capital budgeting problem involves a single objective function deterministic approach to the allocation of limited resources among available investment opportunities. This approach differs greatly from most real-world capital budgeting problems. Actual resource allocation distribution procedures involve an analysis which is by necessity nondeterministic and sensitive to numerous conflicting interests. Due in part to this divergence between the traditional mathematical model of the capital budgeting problem and the necessities of real-world decision making, a significant amount of discussion has been generated concerning the traditional approach and its applicability to today's complex decision-making procedures [1-11].

The solution to the capital budgeting problem obtained in a model which seeks a compromise from among the numerous objective functions which represent the decision situation is a more viable methodology to characterize today's complex decision-making situations [5, 10, 12, 13]. Rather than a single objective function model of the capital budgeting problem, the general multiple objective function model takes on the following form:

$$\text{Maximize} \quad \sum_{j=1}^{n} r_{kj}\, x_j \qquad k = 1, 2, \ldots, K$$

$$\text{Subject to} \quad \sum_{j=1}^{n} c_{ij}\, x_j \leq b_i \qquad \forall i$$

$$x_j = (0,1) \quad ,$$

where the terms are defined as:

$$x_j = \begin{cases} 1 & \text{if the } j^{th} \text{ alternative is selected} \\ 0 & \text{if the } j^{th} \text{ alternative is not selected,} \end{cases}$$

r_{kj} = return on objective k from alternative j ,

c_{ij} = requirement of resource i by alternative j , and

b_i = limitation of resource i .

Many multiple objective optimization techniques have been employed in the solution of this problem; these were discussed in Chapter 2.

5.2 Fuzzy Linear Integer Programming/Exchange Heuristic Algorithm

An algorithm is developed which combines the principles of fuzzy linear programming and Petersen's [42] exchange heuristic to solve the multiple objective capital budgeting problem. The algorithm is intended to solve the following capital budgeting problem:

$$\text{Maximize} \quad \sum_{j=1}^{n} r_{kj}\, x_j \qquad k = 1, 2, \ldots, K$$

$$\text{Subject to} \quad \sum_{j=1}^{n} c_{ij}\, x_j \leq b_i \qquad \forall i$$

$$x_j = (0,1) \quad ,$$

where the terms are defined previously.

The algorithm is a three-phase solution technique which incorporates an interactive process between the analyst and the decision maker in Phase I. In Phase II, a fuzzy linear integer problem is solved. Phase III, the exchange heuristic, is utilized if a 0,1 solution was not obtained in Phase II.

5.3 The Algorithm

5.3.1 Phase I: Determination of aspiration levels and the lowest admissible values. Phase I of the algorithm is intended to be an interactive process between the analyst and the decision maker. In this phase, K successive linear programming problems are solved, where K is the number of fuzzy objectives. The constraint set is to remain constant throughout the evaluations. In this manner, each objective function yields the highest attainable value possible. This value will be referred to as the Aspiration Level.

The lowest admissible value for each function is determined from the programs which yield the aspiration levels for the other K-1 functions. The value determined to be the lowest admissible value when subtracted from the aspiration level yields the allowable tolerance interval for each objective function.

The calculated values for the aspiration levels, lowest admissible values, and the tolerance intervals should then be reviewed by the decision maker. It rests with the decision maker to provide the analyst with the values to continue the algorithm in Phase II. This interactive process is critical to the fundamental concept of fuzzy programming, that the theory of fuzzy sets combines the quantitative aspects of optimization with the judgmental abilities of decision makers.

The programming procedure utilized to complete this phase is the IBM MPSX Linear Programming technique. Appendix A discusses the IBM MPSX system in greater detail.

5.3.2 Phase II: Determination of a fuzzy linear integer programming solution. In Phase II, a fuzzy transformation is carried out on each fuzzy function, and a linear integer programming problem is solved to maximize the value of the membership function.

The fuzzy transformation depends on the type of function under consideration. The three possibilities are shown in Table V.1. The $\overline{d}_i$ and $\underline{d}_i$ are the selected upper and lower bounds of the tolerance interval specified by the decision maker. Graphically, these three functions are shown in Figures 7, 8, and 9.

Table V.1. Fuzzy Transformations

Type	Objective	
I.	Equal or exceed b_i	$\lambda \leq 1 - \frac{b_i' - (Zx)_i}{\underline{d}_i}$
II.	Equal or less than b_i	$\lambda \leq 1 - \frac{(Zx)_i - b_i'}{\overline{d}_i}$
III.	Equal b_i	a. $\lambda \leq 1 - \frac{b_i' - (Zx)_i}{\underline{d}_i}$ and b. $\lambda \leq 1 - \frac{(Zx)_i - b_i'}{\overline{d}_i}$

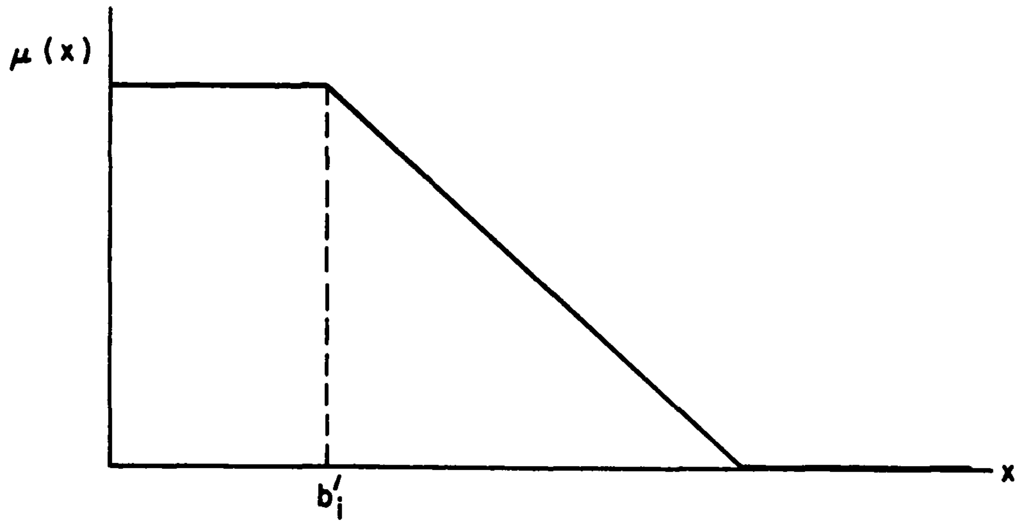


Figure 7. Membership Function for Type I Objective

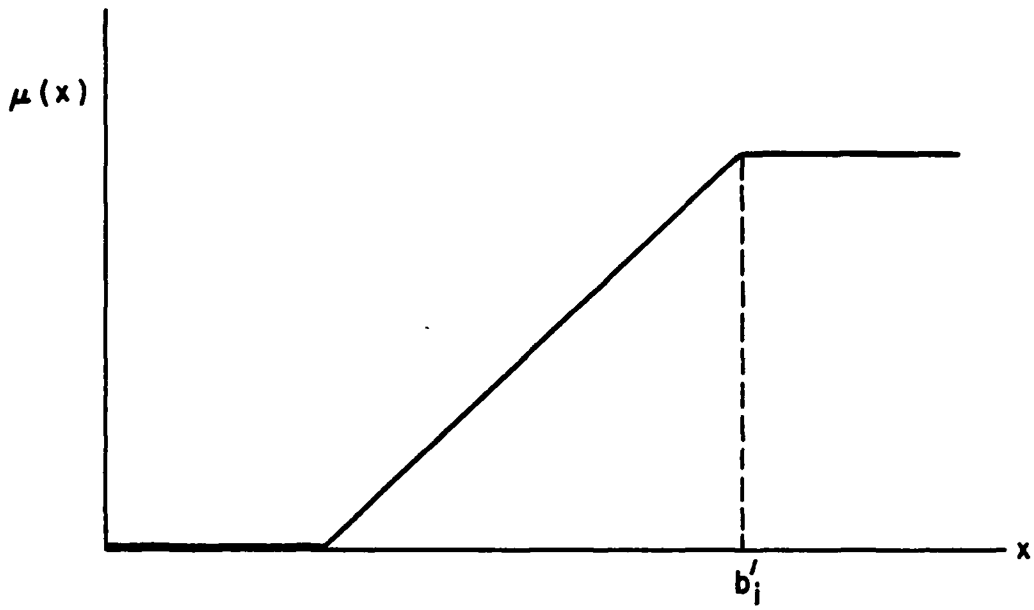


Figure 8. Membership Function for Type II Objective

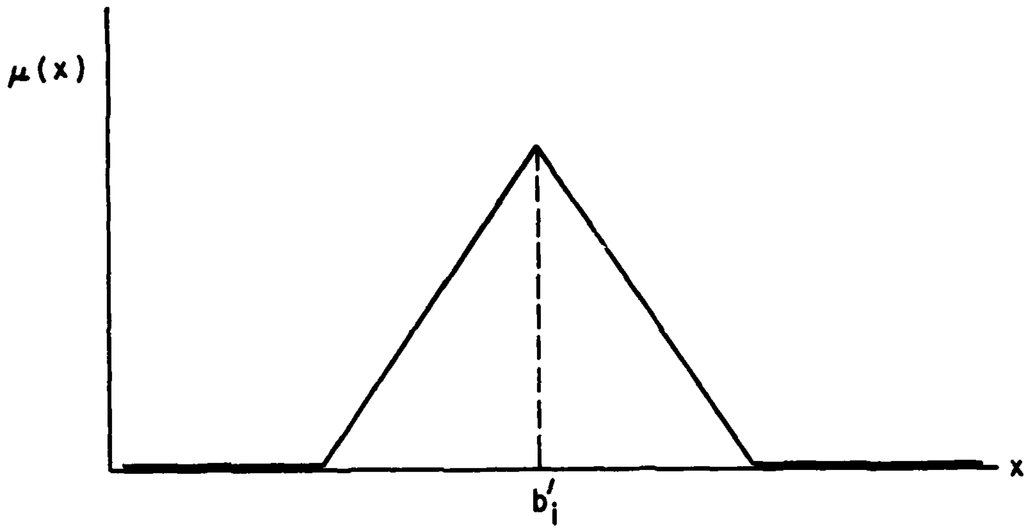


Figure 9. Membership Function for Type III Objective

The fuzzy linear integer programming problem formulation typically may be expressed as follows:

Maximize λ

$$\text{Subject to } \lambda \leq 1 - \frac{b_i' - (Zx)_i}{d_i} \quad , \quad i = 1, \ldots, K$$

$$\text{and } \quad (Ax)_i \leq b_i \quad ,$$

$$x_j \leq 1 \quad \forall j \quad ,$$

where $(Ax)_i$ is the set of rigid constraint functions, and each variable has an upper bound of 1.0 .

If the solution to this linear programming problem is satisfactory to satisfy the 0-1 restrictions, then the algorithm terminates; otherwise, proceed to Phase III.

5.3.3 <u>Phase III: Determination of an exchange heuristic 0-1 solution</u>. The exchange heuristic, a modified form of Petersen's [42], is composed of three major steps:

i. Determination of an initial solution.

ii. Determination of a fitback solution.

iii. Utilize exchange operations progressively to improve the solution so as to finally achieve at least a local optimum.

<u>Determination of an Initial Solution</u> - The initial solution is obtained after ranking each variable based on the value of the ratio T_j/R_j given n variables and m objective functions, where T_j is the summation of the coefficient

values of each variable in the fuzzy objectives, and

R_j is defined as $\sum_{i=1}^{n} (c_{ij}/b_i) \qquad j = 1, 2, \ldots, n$.

The variables with the highest values of the ratio T_j/R_j are placed at the top of the ranking list. Variables are rejected from the bottom of the list until the rejection of the K^{th} variable causes satisfaction of $\sum_{j=1}^{K-1} c_{ij} \leq b_i$ for all rigid constraints. The initial solution is comprised of those variables ranked 1 through K-1 .

Determination of a Fitback Solution - In general, following the selection of an initial solution, there will be some degree of slack for each constraint. The fitback solution selects from the initially nonselected variables ranked $K + 1$ to n , one or more that can be included with the selected variables without violating any constraint.

Exchange Operations - The alternatives in the sets of selected and nonselected variables are ranked according to their T_j value. In the set of selected variables, the variables are ranked starting with the lowest value first, while in the set of nonselected variables, variables are ranked with the highest value first.

The search procedure is a two-step process. For each exchange, it is determined if the exchange under consideration would cause an improvement in the membership

function. If an improvement is noted, then the feasibility of the exchange is examined.

The set of exchanges is divided into two groups. The first search consists of the 2/1, 1/1, and 1/2 exchanges, while the second search considers 3/1, 3/2, and 3/3 exchanges. In each case, the first number refers to the number of variables selected from the set of nonselected variables.

The sequencing of the variables in the sets of selected and nonselected variables is performed to reduce the number of searches necessary to obtain a solution. The sequence allows for the examination of the most profitable exchanges first. Then, if an exchange is advantageous, the search is reduced due to dominance. In ordering the sets of selected and nonselected variables, the search proceeds naturally from the most advantageous exchanges to least advantageous exchanges.

5.4 Example of Three-Phase Algorithm

The three-phase algorithm is most easily explained via an example. Consider a problem in which the decision-making situation is characterized by two fuzzy objective functions and three rigid constraint functions. Assume this decision has the following problem formulation:

$$\text{Maximize} \quad Z_1 = 3x_1 + 5x_2 + 5x_3 + x_4$$

$$\text{Maximize} \quad Z_2 = x_1 + x_3 + x_4$$

$$\begin{aligned} \text{Subject to } 2x_1 + x_2 + 3x_3 + x_4 &\le 6 \\ x_1 + 2x_2 + 4x_3 + 2x_4 &\le 5 \\ 3x_1 + 2x_2 + x_3 + x_4 &\le 4 \\ x_j &= (0,1) \quad . \end{aligned}$$

5.4.1 Determine the Aspiration Level and Lowest Admissible Value for each objective. To calculate the aspiration level of the objective functions, the optimization technique of linear programming is utilized. Solving a linear programming problem to maximize each objective function subject to the same set of constraint functions yields the highest attainable value of the solution or the aspiration level. Thus, for the example:

(a) $$\text{Maximize} \quad Z = 3x_1 + 5x_2 + x_4$$

$$\begin{aligned} \text{Subject to } 2x_1 + x_2 + 3x_3 + x_4 &\le 6 \\ x_1 + 2x_2 + 4x_3 + 2x_4 &\le 5 \\ 3x_1 + 2x_2 + x_3 + x_4 &\le 4 \end{aligned}$$

Solution: $Z_1 = 9.55$

$x_1 = 0.45$ $\quad$ $x_3 = 0.64$

$x_2 = 1.00$ $\quad$ $x_4 = 0.0$.

(b) Maximize $Z_2 = x_1 + x_3 + x_4$

Subject to
$$2x_1 + x_2 + 3x_3 + x_4 \leq 6$$
$$x_1 + 2x_2 + 4x_3 + 2x_4 \leq 5$$
$$3x_1 + 2x_2 + x_3 + x_4 \leq 4$$

Solution: $Z_2 = 5.36$

$x_1 = 0.82$ $\quad x_3 = 0.55$

$x_2 = 0.0$ $\quad x_4 = 1.00$.

To calculate the lowest admissible value for each of n objective functions, evaluate each objective function with the other $n - 1$ linear programming solution programs. Select as the lowest admissible value for each objective function the minimum resulting evaluation. Thus, for the example:

(a) Evaluate objective function Z_1 with the program obtained in Item (b) of the determination of the aspiration level.

$$Z_1\big|_{(0.82,0,0.55,1.0)} = 3x_1 + 5x_2 + 5x_3 + x_4$$
$$Z_{1(LAV)} = 6.21 \quad .$$

(b) Evaluate objective function Z_2 with the program obtained in Item (a) of the aspiration level.

$$Z_2\big|_{(0.45,1.0,0.64,0)} = x_1 + x_3 + x_4$$
$$Z_{2(LAV)} = 1.09 \quad .$$

The results of Phase I of the algorithm are summarized in Table V.2.

Table V.2. Summary of Calculations in Phase I

Objective Function	Aspiration Level	Lowest Admissible Value	Tolerance Interval
Z_1	9.55	6.21	3.34
Z_2	5.36	1.09	4.27

5.4.2 Phase II: Fuzzy linear integer programming formulation. The initial step in the fuzzy linear integer programming problem formulation is to determine the type of objective function and transform the objective function as appropriate. The fuzzy transformations were shown in Table V.1. Thus, for the example:

Since both fuzzy functions are Type I functions, the transformations are as follows:

$$Z_1: \quad \lambda \leq 1 - \frac{[9.55 - (3x_1 + 5x_2 + 5x_3 + x_4)]}{3.34}$$

$$\lambda \leq 1 - (2.85 - 0.8982x_1 - 1.497x_2 - 1.497x_3 - 0.299x_4)$$

$$\lambda \leq - 1.85 + 0.8982x_1 + 1.497x_2 + 1.497x_3 + 0.299x_4)$$

$$Z_2: \quad \lambda \leq 1 - \frac{[5.36 - (x_1 + x_3 + 4x_4)]}{4.27}$$

$$\lambda \leq 1 - (1.255 - 0.2342x_1 - 0.2342x_3 - 0.9367x_4)$$

$$\lambda \leq - 0.255 + 0.2342x_1 + 0.2342x_3 + 0.9367x_4 \quad .$$

The formulation as a fuzzy linear integer programming problem is:

$$
\begin{array}{ll}
\text{Maximize} & \lambda \\
\text{Subject to} & \lambda \leq -1.85 + 0.8982x_1 + 1.497x_2 + 1.497x_3 + 0.299x_4 \\
& \lambda \leq -0.255 + 0.2342x_1 + 0.2342x_3 + 0.9367x_4 \\
& 2x_1 + x_2 + 3x_3 + x_4 \leq 6 \\
& x_1 + 2x_2 + 4x_3 + 2x_4 \leq 5 \\
& 3x_1 + 2x_2 + x_3 + x_4 \leq 4 \\
& x_j = (0,1) \quad .
\end{array}
$$

$$
\begin{array}{lll}
\text{Solution:} & \lambda = 0.21 & \\
& x_1 = 1.0 & x_3 = 1.0 \\
& x_2 = 0.0 & x_4 = 0.0 \quad .
\end{array}
$$

The solution is in the form (0,1) ; however, Phase III will be utilized to illustrate the exchange heuristic.

5.4.3 Phase III: Exchange Heuristic solution. The first step in this Exchange Heuristic approach is to set up the problem in the standard form as described previously (Section 5.2). In the example under consideration, this formulation is as follows:

$$
\begin{array}{ll}
\text{Maximize} & \lambda \\
\text{Subject to} & 2x_1 + x_2 + 3x_3 + x_4 \leq 6 \\
& x_1 + 4x_3 + 4x_3 + 2x_4 \leq 5 \\
& 3x_1 + 2x_2 + x_3 + x_4 \leq 4 \\
& \lambda - 0.8982x_1 - 1.497x_2 - 1.497x_3 - 0.299x_4 \leq -1.85 \\
& \lambda - 0.2342x_1 - 0.2342x_3 - 0.9367x_4 \leq -0.255 \\
& x_j = (0,1) \quad .
\end{array}
$$

The second step is to determine an initial solution. The variables comprising the initial solution are determined by ranking each variable based on the ratio T_j/R_j . T_j is defined as the summation of the coefficients of each variable in the fuzzy objective functions. In the example, there are two fuzzy objectives. Thus, in the example, the calculation of the T_j values is as follows:

Table V.3. Calculation of T_j Values

Variable	Fuzzy Objective 1	Fuzzy Objective 2	$T_j = \sum c_{ij}$
x_1	-0.8982	-0.2342	-1.1324
x_2	-1.497	0.0	-1.497
x_3	-1.497	-0.2342	-1.7312
x_4	-0.2990	-0.9367	-1.2357

The R_j values are calculated by evaluating the ratio of the coefficients of both the fuzzy objective functions and the rigid constraint functions to the appropriate b_i values. In the example problem, the calculation of the R_j values is as follows:

Table V.4. Calculation of R_j Values

Variable	R_j	R_j Values
x_1	$\frac{-0.8982}{-1.85} + \frac{-0.2342}{-0.255} + \frac{1}{5} + \frac{3}{4} + \frac{2}{6}$	2.69
x_2	$\frac{-1.497}{-1.85} + 0 + \frac{2}{5} + \frac{2}{4} + \frac{1}{6}$	1.876
x_3	$\frac{-1.497}{-1.85} + \frac{-0.2342}{-0.255} + \frac{4}{5} + \frac{1}{4} + \frac{3}{6}$	3.27
x_4	$\frac{-0.2990}{-1.85} + \frac{-0.9367}{-0.255} + \frac{2}{5} + \frac{1}{4} + \frac{1}{6}$	4.65

The ratio T_j/R_j is calculated from the results obtained in Table V.3 and Table V.4. In the example problem, the T_j/R_j values are:

Table V.5. Calculation of T_j/R_j Values

Variable	T_j	R_j	T_j/R_j
x_1	-1.1324	2.69	-0.420
x_2	-1.497	1.876	-0.798
x_3	-1.7339	3.27	-0.530
x_4	-1.2357	4.65	-0.266

The initial solution may be calculated by ranking the n variables according to their respective T_j/R_j values. Variables with the highest values of the T_j/R_j ratio are placed at the top of the ranking list. Variables are rejected from the bottom of the list until the rejection of the K^{th} variable causes satisfaction of $\sum_{j=1}^{k-1} c_{ij} \leq b_i$

for all rigid constraints. The initial solution is comprised of those variables ranked 1 through K-1 . The initial ranking of the variables is as follows:

Table V.6. Initial Ranking of Variables

Variable	Initial Ranking
x_1	2
x_2	4
x_3	3
x_4	1

The initial solution may be determined as follows:

Table V.7. Determination of the Initial Solution

Objective Function	Functional Evaluation {4,1,3,2}	{4,1,3}	{4,1}		
1	7*	6	3	Initial Solution	{4,1}
2	9*	7*	3	Set of Selected Variables:	{4,1}
3	7*	5*	4	Set of Nonselected Variables:	{2,3}
4	-4.19	-0.269	-1.19	Value of Membership Function:	0.0
5	-1.41	-1.40	-1.17		

*Indicates constraint functions that are not satisfied.

The third step in the exchange heuristic is to determine a fitback solution. The fitback solution selects from the set of initially nonselected variables one or more than can be included with the selected variables without violating any constraint. Table V.8 displays the calculation of the fitback solution.

Table V.8. Determination of Fitback Solution

Objective Function	Functional Evaluation {4,1,3}	{4,1,2}	
1	6	4	Since at least one constraint is violated in each possible fitback solution, therefore, the fitback solution is the same as the initial solution, {4,1}.
2	7*	5	
3	5*	6*	
4	-2.69	-2.69	
5	-1.40	-1.40	

*Indicates constraint functions that are not satisfied.

The fourth step consists of utilizing exchange operations progressively to improve the solution. After each exchange, the feasibility of the exchange is examined. If the exchange is feasible, the possible improvement in the membership function is examined. If an improvement is not noted, then proceed to the next exchange possibility. Table V.9. displays the search procedure examining the possible exchanges.

Table V.9. First Search Exchange Procedure

First Search

List of Selected Variables: {4,1}

List of Nonselected Variables: {2,3}

Attempted Exchange		λ	λ max	Selected Variables
(2/1)			0.0	{4,1}
(2,3) for 4	Infeasible Exchange	--	0.0	{4,1}
(2,3) for 1	Infeasible Exchange	--	0.0	{4,1}
(1/1)				
(2) for 4	Infeasible Exchange	--	0.0	{4,1}
(2) for 1	Infeasible Exchange	--	0.0	{4,1}
(3) for 4	Advantageous/ Feasible	0.21	0.21	{3,1}

Repeat First Search

List of Selected Variables: {3,1}

List of Nonselected Variables: {2,4}

The algorithm utilizes a second pass through the list of non-selected variables once a favorable exchange has been noted. In the example problem, the second search produced no exchange advantageous to the membership function. Therefore, the final solution to the example is:

Set of Selected Variables: $\{3,1\}$

Value of Membership Function: $\lambda = 0.21$

5.5 Example Number 2, Project Selection Example

In order to illustrate a capital budgeting problem in which the decision situation is program or project selection, the following example is presented. In this example, the decision maker has specified firm values for the aspiration levels and acceptable ranges of admissibility for each fuzzy function [6].

A systems engineer has to design an integrated system composed of three subsystems, designated A, B, and C. Three systems have been proposed for Subsystem A, four for Subsystem B, and three for Subsystem C. Four attributes were established by management to guide in the selection of the subsystems. These are weight, development costs, estimated reliability, and power requirements. Table V.10 summarizes the attribute characteristics for each proposed candidate.

Design incompatibilities exist between Candidates A-2 and C-9. Also, due to design features, if Candidate B-7 is selected, then Candidate C-10 must be selected.

Management has established firm values for the aspiration levels and allowable ranges of admissibility. Table V.11 summarizes this information for each attribute.

Table V.10. Attribute Data for Proposed Candidates

	Subsystem A			Subsystem B				Subsystem C		
Attribute	1	2	3	4	5	6	7	8	9	10
Weight (lb.)	32	57	19	95	107	61	48	23	10	15
Cost ($\$10^4$)	120	95	160	64	67	96	119	42	36	70
Reliability	0.97	0.94	0.99	0.89	0.90	0.94	0.96	0.98	0.97	0.99
Power (watts)	21	35	10	60	83	27	50	12	7	16

Table V.11. Management Specified Attribute Data

Attribute	Aspiration Level	Lowest Admissible Value	Highest Admissible Value
Weight (lb.)	150	120	165
Cost ($\$10^4$)	195	260	- -
Power (watts)	100	70	110

Mathematically, this problem may be formulated as a system of linear equations. The reliability constraint may be transformed to a linear equation via the transformation $Z = (Y)^x = Z\ell nY$:

$$Z_1 = 32x_1 + 57x_2 + 19x_3 + 95x_4 + 107x_5 + 61x_6 + 48x_7 + 23x_8 + 10x_9 + 15x_{10}$$

$$Z_2 = 120x_1 + 95x_2 + 160x_3 + 64x_4 + 67x_5 + 96x_6 + 119x_7 + 42x_8 + 36x_9 + 70x_{10}$$

$$Z_3 = 21x_1 + 35x_2 + 10x_3 + 60x_4 + 83x_5 + 27x_6 + 50x_7 + 12x_8 + 7x_9 + 16x_{10}$$

Subject to

$$(0.97)^{x_1}(0.94)^{x_2}(0.99)^{x_3}(0.89)^{x_4}(0.90)^{x_5}(0.94)^{x_6}$$

$$(0.96)^{x_7}(0.98)^{x_8}(0.97)^{x_9}(0.99)^{x_{10}} \geq 0.85$$

$$x_1 + x_2 + x_3 = 1$$

$$x_4 + x_5 + x_6 + x_7 = 1$$

$$x_8 + x_9 + x_{10} = 1$$

$$x_2 + x_5 = 1$$

$$x_7 - x_{10} = 0$$

$$x_j = (0,1)$$

Transforming the problem into its fuzzy equivalent, the problem formulation is as follows:

Maximize λ

Subject to

$$\lambda + 1.84x_1 + 1.46x_2 + 2.46x_3 + 0.98x_4 + 1.03x_5 + 1.47x_6$$
$$1.83x_7 + 0.64x_8 + 0.55x_9 + 1.08x_{10} \leq 4.0$$

$$\lambda - 1.067x_1 - 1.9x_2 - 0.64x_3 - 3.17x_4 - 3.56x_5 - 2.03x_6$$
$$- 1.6x_7 - 0.76x_8 - 0.33x_9 - 0.5x_{10} \leq -4.0$$

$$\lambda + 2.13x_1 + 3.8x_2 + 1.26x_3 + 6.34x_4 + 7.13x_5 + 4.06x_6$$
$$+ 3.2x_7 + 1.53x_8 + 0.67x_9 + 1.0x_{10} \leq 11.0$$

$$\lambda - 0.70x_1 - 1.16x_2 - 0.34x_3 - 2.0x_4 - 2.77x_5 - 0.9x_6$$
$$- 1.66x_7 - 0.4x_8 - 0.23x_9 - 0.53x_{10} \leq -2.33$$

$$\lambda + 2.1x_1 + 3.5x_2 + 1.0x_3 + 6.0x_4 + 8.3x_5 + 2.7x_6 + 5.0x_7$$
$$+ 1.2x_8 + 0.7x_9 + 1.6x_{10} \leq 11.0$$

$$\lambda - 0.032x_1 - 0.062x_2 - 0.01x_3 - 0.117x_4 - 0.105x_5 - 0.062x_6$$
$$- 0.041x_7 - 0.020x_8 - 0.030x_9 - 0.01x_{10} \geq -0.1625$$

$$1.0x_2 + 1.0x_5 = 1.0$$

$$1.0x_1 + 1.0x_2 + 1.0x_3 = 1.0$$

$$1.0x_4 + 1.0x_5 + 1.0x_6 + 1.0x_7 = 1.0$$

$$1.0x_8 + 1.0x_9 + 1.0x_{10} = 1.0$$

$$1.0x_7 - 1.0x_{10} = 0$$

$$x_j = (0,1)$$

The solution to the fuzzy linear programming problem is:

$\lambda = 0.1344$

x_1	= 0.0	x_6	= 1.0
x_2	= 1.0	x_7	= 0.0
x_3	= 0.0	x_8	= 1.0
x_4	= 0.0	x_9	= 0.0
x_5	= 0.0	x_{10}	= 0.0 .

The solution is in the required 0-1 form, the exchange heuristic or Phase III is not necessary. Therefore the decision is summarized as follows:

$\lambda = 0.1334$

List of Selected Projects: {2,6,8}

List of Nonselected Projects: {1,3,4,5,7,9,10} .

The rigid constraints were all satisfied. The fuzzy objectives were calculated as follows:

Table V.12. Example 2, Attribute Satisfaction

Attribute	Calculated Value
Weight (lb.)	141
Cost ($\$10^4$)	\$233
Power (watts)	74

5.6 Analysis and Discussion of Results

The decision problem presented in Section 5.5 was originally solved via a "multirisk" programming model [6]. This analysis concept is based on the determination of the alternative decision solutions which minimize the probabilities that the decision maker's objectives and constraints will not be satisfied. The "best" subset of m decision alternatives from a possible set of n candidates is selected such that the problem's objectives and constraints are satisfied with minimum risk.

The multirisk programming model is designed to solve multiple criteria decision problems, assuming that decision makers strive to achieve or satisfy goals rather than attempting to optimize them. The decision maker may incorporate the concept of "fuzziness" [66] into the analysis by specifying a range of deviation allowable for the goals and constraints of the problem.

The multirisk programming model is a stochastic analysis technique for solving multiple criteria decision problems. Problem formulation may include both rigid and stochastic goals and constraints. The rigid goals and constraints have deterministic coefficients, while the stochastic goals and constraints have stochastic coefficients. The model assumes that the range of deviation is a random variable, while the stochastic parameters of the objectives and constraints are normally distributed independent random variables. The "best" solution to the multirisk programming model is the program which minimizes the risk of not achieving the goals and constraints of the problem, or maximizes the probability that the goals and constraints are achieved.

In the multirisk programming analysis of the project selection example (Section 5.5), uncertainties were incorporated for each coefficient. The attribute data for the proposed candidates is shown in Table V.13.

The multirisk programming model employs an enumerative search to identify the "best" solution. Table V.14 displays the results of the multirisk programming model and the fuzzy linear integer programming/exchange heuristic model. Both fuzzy linear integer programming/exchange heuristic model and the multirisk programming model of multiple criteria decision making provide effective mathematical modeling techniques for the analysis of management decisions which are fuzzy in nature.

The problem formulation of the fuzzy linear programming/exchange heuristic model was presented in Section 5.2. This multiple objective analysis provides the decision maker "leeway" in modeling phenomena of a vague or ill-defined nature. This "leeway" in the model is a result of utilizing fuzzy sets to describe those objective functions and constraints that are imprecisely defined. Through the use of a fuzzy set operator, the "min" operator, an optimal decision in the fuzzy environment is obtained. This optimal decision is defined as the point which maximizes the membership function of the fuzzy set formed through the intersection of those fuzzy sets representing the various objective functions and constraints. The exchange heuristic in the model seeks to obtain the best attainable integer solution given that the optimal point defined above is not integer valued.

Table V.13. Attribute Data for Proposed Candidates

	Subsystem A			Subsystem B				Subsystem C		
Attribute	1	2	3	4	5	6	7	8	9	10
Weight (lb.)	32	57	19	95	107	61	48	23	10	15
Cost ($\$10^4$)	120	95	160	64	67	96	119	42	36	70
Reliability	0.97	0.94	0.99	0.89	0.90	0.94	0.96	0.98	0.97	0.99
Power (watts)	21	35	10	60	83	27	50	12	7	16

Uncertainties in weights = $\pm$ 10%.

Uncertainties in costs = $\pm$ 15%.

Reliabilities are assumed constant.

Uncertainties in power requirements = $\pm$ 5%.

Table V.14. Comparison of Results

System Attribute	Fuzzy LP/Heuristic Model	Multirisk Programming Model	
		Expected Value	Standard Deviation
Weight	141 lbs.	141 lbs.	4.3 lbs.
Cost	$2,330,000	$2,330,000	$106,000
Reliability	0.866	0.866	0.0
Power	74 watts	74 watts	1.1 watts

One practical advantage of modeling with either of these techniques is that the decision situation does not have to be defined in a precise manner. In the fuzzy programming approach, the decision maker specifies ranges of acceptability for those objective functions and constraints represented by fuzzy sets. In the multirisk programming model, the decision maker specifies ranges of deviation. In both models, the decision maker is given greater flexibility than would be available in a classical mathematics approach.

A major advantage of fuzzy linear programming is the ease with which it can be formulated and solved on numerous mathematical programming systems. The exchange heuristic and the multirisk programming techniques both require utilizing a specific computer program which may not be readily available.

The major advantage of the multirisk programming model is its ability to analyze multiple criteria decision problems which are characterized by nondeterministic coefficients for the various objective functions. This model provides the decision maker leeway in defining his aspiration levels, as well as in stating precisely the terms of the objective functions.

The enumerative search technique employed in the multirisk programming model is impractical for large scale problems both in computer storage requirements and necessary CPU time [6]. The fuzzy linear programming model utilizes whatever mathematical programming system that is available to the user to solve mixed integer linear programming problems. The computer storage requirement is, therefore, programming package dependent, as is the CPU time necessitated.

5.7 Computer Program

A computer program implementing the fuzzy linear integer programming and the exchange heuristic phases of the algorithm was developed in FORTRAN IV for the IBM 370/3033 computer system. Standard FORTRAN language was employed to permit relative ease of adaptation of the computer model for use on other computer systems. The amount of internal storage necessitated on the IBM 370/3033 was 280,000 bytes. The amount of storage necessary is due to the requirements of the MPSX system. The computer program is currently dimensioned for the comparison of twenty-five alternatives. This program size could be enlarged by redimensioning the program not to exceed the MPSX variable limit. The exchange heuristic program is capable of solving problems of size one hundred and fifty constraints with one hundred and fifty decision variables with 280,000 bytes of storage. The CPU time to execute Example 1 was 2 seconds, while 3 seconds of CPU time was required for Example 2.

5.8 Computer Code Description

The fuzzy linear integer programming computer code and the exchange heuristic computer code are listed in Appendix B along with the definition of all input data. The computer codes will be described in three sections:

i. Fuzzy Linear Programming Transformation program.

ii. IBM MPSX/Mixed Integer Programming Control program.

iii. Exchange Heuristic program.

The fuzzy linear integer programming transformation program is composed of a single main program. This program reads the input data

and transforms the objective functions into their fuzzy transformation as appropriate. Temporary data sets are created to be used as data input for the MPSX/Mixed Integer Programming Optimization Technique or the exchange heuristic program as appropriate.

The MPSX/Mixed Integer Programming Control program is an advanced usage example [63] of the IBM MPSX/MIP technique. This control program optimizes the continuous problem, then solves the mixed integer problem. A more in-depth discussion of the MPSX mathematical programming system is presented in Appendix A.

The exchange heuristic code consists of a main program and seven subroutines. The code is a modified version of Petersen's [42] heuristic algorithm, with subsequent modifications by Bouillot and Smith [64, 65]. The description of this code is as follows:

Main Program. The main program reads the input data from the temporary data set created in the fuzzy linear programming transformation program. This program calculates the R_j values and the ratio T_j/R_j for each variable. It maintains the list of selected and nonselected variables and determines those exchanges to be executed. It calls the various subroutines in the proper sequence required to conduct the exchange operations. It formats and writes all output data as required.

Subroutine Rank. This subroutine initially ranks the variables in both the sets of selected and nonselected variables.

Subroutine Impvmt. This subroutine maintains the best solution achieved that is both advantageous and feasible.

Subroutine Feasbl. This subroutine examines the feasibility of the exchange under consideration.

Subroutine Exchge. This subroutine executes the exchange operations.

Subroutine Achvmt. This subroutine calculates the gain in the membership function as a result of an exchange.

Subroutine OBJ. This subroutine evaluates the various fuzzy objective functions to determine the lambda value.

Subroutine FTOBJ. This subroutine calculates the fitback solution.

CHAPTER 6

SUMMARY, CONCLUSIONS, AND SUGGESTIONS FOR FURTHER RESEARCH

In this chapter, the work presented in this thesis is summarized, and a few conclusions are drawn about the fuzzy capital budgeting model and the general applicability of fuzzy programming.

6.1 Summary and Conclusions

The model of the capital budgeting problem explored in this work is a combined application of the models developed by Zimmermann [11] and Petersen [42]. The fuzzy linear integer programming approach to management decisions is designed to study decision problems involving multiple goals and constraints, some of which are of a vague or ill-defined nature. The method is founded on the theory of fuzzy sets. Fuzzy sets are utilized to model phenomena of an ill-defined nature which cannot be described adequately in classical mathematical terms. The analysis seeks to permit the human mind to utilize its capabilities to the fullest extent, while utilizing the computational efficiency of the computer to perform those operations which the human mind cannot adequately accomplish. This analysis allows the individual decision maker to make small judgmental decisions (what the human mind does best), while allowing the computer to solve large linear programming problems incorporating these judgmental decisions. Since the solution to the fuzzy linear integer programming problem may not be in the form (0,1), a modified form of Petersen's exchange heuristic for the capital

budgeting problem was employed. This exchange heuristic seeks the "best" attainable solution, given that no integer solution exists.

The model is designed to solve the general problem in which the "best" subset of m alternatives is selected from a candidate set of n possible decision alternatives, such that the membership function of the fuzzy set of the decision is maximized. The model provides a great deal of flexibility to the user in formulating problems for analysis. The availability of solution algorithms and computer solution systems which are readily compatible with the fuzzy linear integer programming problem formulation allows the user to realize a computational solution with ease.

This analysis has applicability for a broad range of decision problems involving the selection of entities from among numerous alternative possibilities, such as equipment purchases, route selection, or investment selection. The example presented in Chapter 5 successfully analyzed the selection of alternative subsystems in achieving system design requirements, while satisfying stated cost restrictions.

6.2 Suggestions for Further Research

The work described in this thesis can be extended in several different directions. The integer solution technique utilized in this work was the MPSX/Mixed Integer Programming System. Although this mathematical programming system readily yields a solution to the problem, the availability of MPSX is not universal. The development and use of an integer programming computer code in standard FORTRAN would greatly enhance the ease with which the model could be adapted to other computers.

The popularity of multiple objective analysis via goal programming could be the catalyst of another extension. The formulation of the fuzzy objectives and constraints into a goal programming analysis would be an extremely interesting development. Goal programming is an extremely robust optimization technique, which is viewed as a practical and natural representation of a wide variety of real-world problems. In combining the fuzzy programming approach of optimizing humanistic systems which are by nature vague and ill-defined, with the practicality of goal programming, an optimization methodology may result which presents a realistic perspective of management decision making.

Field experimentation with fuzzy programming models of management decision making would be desirable. Zimmermann [58] has conducted numerous experiments to analyze the viability of modeling decision makers via the concepts inherent in fuzzy programming. Applications of fuzzy programming include personnel management and determination of credit worthiness in the banking industry [58], media selection [11], and the sizing of a truck fleet [9].

The fuzzy set operator used in the fuzzy linear integer programming model is the "min" operator. Zimmermann and Hamacher [58] have experimented on the applicability of other operators in the optimization of management decisions. These operators include the product operator, the algebraic sum operator, the max operator, both the arithmetic and geometric mean operators, and the gamma operator.

Certainly, many other areas for further research exist. However, these few are listed to provide the reader some idea as to where additional research might begin.

REFERENCES CITED

[1] Evans, J. P. and Steuer, Ralph E., "A Revised Simplex Method for Linear Multiple Objective Programs," Mathematical Programming, Vol. 5, No. 1, 1973, pp. 54-72.

[2] Easton, A., Complex Managerial Decisions Involving Multiple Objectives, John Wiley and Sons, Inc., New York, 1973.

[3] Ignizio, J. P., "A Review of Goal Programming: A Tool for Multiobjective Analysis," The Journal of the Operational Research Society, Vol. 29, No. 11, 1978, pp. 1109-1119.

[4] Ignizio, J. P., "An Approach to the Capital Budgeting Problem with Multiple Objectives," The Engineering Economist, Vol. 21, No. 4, 1976, pp. 259-272.

[5] Ignizio, J. P., Goal Programming and Extensions, D. C. Heath, Lexington, MA, 1976.

[6] Odom, Pat R., "A Risk Minimization Approach to Multiple Criteria Decision Analysis," an unpublished Ph.D. Dissertation, The University of Alabama in Huntsville, 1976.

[7] Odom, Pat R., Shannon, Robert E., and Buckles, Billy P., "Multi-Goal Subset Selection Problem Under Uncertainty," AIIE Transactions, Vol. 11, No. 1, March 1979, pp. 61-69.

[8] Taylor, Bernard and Keown, Arthur J., "A Goal Programming Application of Capital Project Selection in the Production Area," AIIE Transactions, Vol. 10, No. 1, March 1978, pp. 52-57.

[9] Zimmermann, H. J., "Description and Optimization of Fuzzy Systems," International Journal of General Systems, Vol. 2, 1976, pp. 209-215.

[10] Zimmermann, H. J., "Fuzzy Programming and Linear Programming with Several Objective Functions," International Journal of Fuzzy Sets and Systems, Vol. 1, 1978, pp. 45-55.

[11] Zimmermann, H. J., "Media Selection and Fuzzy Linear Programming," The Journal of the Operational Research Society, Vol. 29, No. 11, 1978, pp. 1071-1084.

[12] Simon, H., Administrative Behavior, Second Edition, The Free Press, New York, 1957.

[13] Lee, S. M., Goal Programming for Decision Analysis, Auerbach Publishers, Inc., Philadelphia, 1972.

[14] Charnes, A. et al., "Note on an Application of a Goal Programming Model for Media Planning," Management Science, Vol. 14, No. 8, April 1968, pp. 431-436.

[15] Gaines, B. R., and Kohout, L. J., "The Fuzzy Decade: A Bibliography of Fuzzy Systems and Closely Related Topics," International Journal of Man-Machine Studies, Vol. 9, No. 1, 1977, pp. 1-68.

[16] Zimmermann, H. J., Theory and Applications of Fuzzy Sets, Institut für Wirtschaftswissenschaften, Aachen, Federal Republic of Germany, 1975.

[17] Kickert, Walter J. M., Fuzzy Theories and Decision Making, Kluwer Boston, Inc., Hingham, MA, 1978.

[18] Yager, R. R., "Multiple Objective Decision-Making Using Fuzzy Sets," International Journal of Man-Machine Studies, Vol. 9, No. 1, 1977, pp. 375-382.

[19] Cochrane, J. L., and Zeleny, M., editors, Multiple Criteria Decision Making, University of South Carolina Press, Columbia, SC, 1973.

[20] Charnes, A., Cooper, W. W., and Miller, M. H., "Application of Linear Programming to Financial Budgeting and Costing of Funds," Journal of Business, January 1959, pp. 20-46.

[21] Benayoun, R., Larichev, O. I., de Montogolfier, J., and Tergny, J., "Linear Programming with Multiple Objective Functions, The Method of Constraints," Automation and Remote Control, Vol. 32, No. 8, 1971, pp. 1257-1264.

[22] Belenson, S. M., and Kapur, K. C., "An Algorithm for Solving Multi-Criteria Linear Programming Problems with Examples," Operational Research Quarterly, Vol. 24, No. 1, 1973, pp. 65-77.

[23] Charnes, A., and Cooper, W. W., Management Models and Industrial Applications of Linear Programming, John Wiley and Sons, Inc., New York, 1961.

[24] Ijiri, Y., Management Goals and Accounting for Control, North Holland Publishing Company, Amsterdam, 1965.

[25] Hawkins, C. A., and Adams, R. A., "A Goal Programming Model for Capital Budgeting," Financial Management, Vol. 3, 1974, pp. 52-57.

[26] Lee, S. M., and Lerro, A. J., "Capital Budgeting for Multiple Objectives," Financial Management, Vol. 3, 1976, pp. 58-66.

[27] Charnes, A., and Nilhaus, R. J., "A Goal Programming Model for Manpower Planning," Management Science Research Report No. 115, Carnegie-Mellon University, 1968.

[28] Kornbluth, J. S. H., "A Survey of Goal Programming," OMEGA, Vol. 1, No. 2, April 1973, pp. 193-205.

[29] Lee, S. M., and Keown, Arthur J., "Integer Goal Programming Model for Capital Budgeting," Seventh Annual Meeting of the American Institute for Decision Sciences, Cincinnati, OH, November 1975.

[30] Contini, B., "A Stochastic Approach to Goal Programming," Operations Research, Vol. 16, May-June 1968, pp. 576-586.

[31] Geoffrion, A., Dyer, J. S., and Feinberg, A., "An Interactive Approach for Multi-Criterion Optimization with an Application to the Operation of an Academic Department," Working Paper No. 176, University of California, Los Angeles, 1971.

[32] Zionts, Stanley, and Wallenius, Jycki, "An Interactive Programming Method for Solving the Multiple Criteria Problem," Management Science, Vol. 22, No. 6, February 1976, pp. 652-663.

[33] Dyer, J. S., "Interactive Goal Programming," Management Science, Vol. 19, No. 1, 1972, pp. 62-70.

[34] Steuer, Ralph E., "An Interactive Multi-Objective Linear Programming Procedure," Management Science, Special Issues on Multi-Criteria Decision Making, 1978.

[35] Steuer, Ralph E., and Schuler, Albert A., "An Interactive Multiple-Objective Linear Programming Approach to a Problem in Forest Management," Operations Research, Vol. 26, No. 2, 1978, pp. 254-269.

[36] Seward, Samuel M., Plane, Donald R., and Hendrick, Thomas E., "Municipal Resource Allocation: Minimizing the Cost of Fire Protection," Management Science, Vol. 24, No. 16, 1978, pp. 1740-1748.

[37] Armstrong, Ronald D., and Willis, E. Cleve, "Simultaneous Investment and Allocation Decisions Applied to Water Planning," Management Science, Vol. 23, No. 10, 1977, pp. 1080-1088.

[38] Nackel, John G., Goldman, Jay, and Fairman, William L., "A Group Decision Process for Resource Allocation in the Health Setting," Management Science, Vol. 24, No. 12, 1978, pp. 1259-1267.

[39] Chiu, Laurence, and Gear, Tong E., "An Application and Case History of a Dynamic R & D Portfolio Selection Model," IEEE Transactions on Engineering Management, Vol. EM-26, No. 1, 1979, pp. 2-7.

[40] Shih, Wei, "A Branch and Bound Method for the Multiconstraint Zero-One Knapsack Problem," The Journal of the Operational Research Society, Vol. 30, 1979, pp. 369-378.

[41] Kepler, C. Edward, and Blackman, A. Wade, "The Use of Dynamic Programming Techniques for Determining Resource Allocations Among Research and Development Projects," IEEE Transactions on Engineering Management, Vol. EM-20, No. 1, 1973.

[42] Petersen, C. C., "A Capital Budgeting Heuristic Algorithm Using Exchange Operations," AIIE Transactions, Vol. 6, No. 2, June 1974, pp. 143-150.

[43] Petersen, C. C., "Solution of Capital Budgeting Problems Having Chance-Constraint; Heuristic and Exact Methods," AIIE Transactions, Vol. 7, No. 2, June 1975, pp. 153-158.

[44] Hillier, F. S., "A Basic Model for Capital Budgeting of Risky Interrelated Projects," The Engineering Economist, Vol. 17, No. 1, 1971, pp. 1-30.

[45] Charnes, A., and Cooper, W. W., "Chance-Constrained Programming," Management Science, Vol. 6, 1959, pp. 73-79.

[46] Healy, W. C., "Multiple Choice Programming," Operations Research, Vol. 12, 1964, pp. 122-138.

[47] Armstrong, R. D., and Balintfy, J. L., "Chance-Constrained Multiple Choice Programming Algorithm," Operations Research, Vol. 23, No. 3, May-June 1975, pp. 494-510.

[48] Odom, Pat R., and Shannon, Robert E., "A Multi-Goal Approach to Capital Equipment Investment," Proceedings of the 1978 Spring Annual Conference of the AIIE, pp. 312-318.

[49] Park, S. Chan, and Theusen, G. J., "Combining the Concepts of Uncertainty Resolution and Project Balance for Capital Allocation Decisions," The Engineering Economist, Vol. 24, No. 2, 1979, pp. 109-127.

[50] Crawford, Dale M., Huntzinger, Bruce C., and Kirkwood, Craig W., "Multi-Objective Design Analysis for Transmission Conductor Selection," Management Science, Vol. 24, No. 16, 1978, pp. 1700-1709.

[51] Keefer, Donald L., "Allocation Planning for R & D with Uncertainty with Multiple Objectives," IEEE Transactions on Engineering Management, Vol. EM-25, No. 1, 1978, pp. 8-14.

[52] MacCrimmon, D. R., "An Overview of Multiple Objective Decision Making," in J. L. Cochrane and M. Zeleny, editors, Multiple Criteria Decision Making, University of South Carolina Press, Columbia, SC, 1973, pp. 18-44.

[53] Plane, D. R., and MacMillian Jr., C., Discrete Optimization: Integer Programming and Network Analysis for Management Decisions, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1971.

As in most integer programming techniques, long computational time is necessary to determine the optimal integer solution.

The IBM program descriptions [63] provide an in-depth discussion of the MPSX system. In addition, numerous sample problems are presented. The examples are detailed from data input through sample outputs. The Mixed Integer Programming program description is especially instructive. It provides sample MPSX control programs and presents an excellent discussion of the mixed integer programming procedure.

[54] Hax, Aranaldo, C., and Wiig, Karl M., "The Use of Decision Analysis in Capital Investment Problems," Sloan Management Review, Vol. 17, Winter 1976, pp. 19-48.

[55] Clarke, T. E., "Decision Making in Technologically Based Organizations: A Literature Survey of Present Practice," IEEE Transactions on Engineering Management, Vol. EM-21, No. 1, February 1974.

[56] Fabozzi, Frank J., "The Use of Operational Research Techniques for Capital Budgeting Decisions, A Sample Survey," The Journal of the Operational Research Society, Vol. 29, No. 1, pp. 39-42.

[57] Zadeh, L. A., "Fuzzy Sets," Information and Control, Vol. 8, 1965, pp. 338-353.

[58] Zimmermann, H. J., "Rational Decision Making in Fuzzy Environments," Working Paper, Institut für Wirtschaftswissenschaften, Aachen, Federal Republic of Germany, 1979.

[59] Hadley, G., Linear Algebra, Addison-Wesley Publishing Co., Inc., Reading, MA, 1961.

[60] Zuckerman, Martin M., Sets and Transfinite Numbers, Macmillan Publishing Co., Inc., New York, 1974.

[61] Kaufmann, A., Introduction to the Theory of Fuzzy Sub-Sets, Academic Press, New York, 1975.

[62] Cooper Leon, and Steinberg, David, Methods and Applications of Linear Programming, W. B. Saunders Company, Philadelphia, PA, 1974.

[63] IBM, Mathematical Programming System--Extended (MPSX), and Generalized Upper Bounding (GUB) Program Description, Program Reference Manual SH20-0968-1, Mathematical Programming System Extended (MPSX) Mixed Integer Programming (MIP) Program Description, Program Reference Manual SH20-0908-1, August 1973.

[64] Bouillot, Andre, and Smith, Harry, "A Heuristic Algorithm for 0-1 Goal Program," an unpublished paper, The Pennsylvania State University, University Park, PA, 1975.

[65] Bouillot, Andre, and Smith, Harry, "A Capital Budgeting Heuristic Algorithm Using Exchange Operations," an unpublished paper, The Pennsylvania State University, University Park, PA, 1975.

[66] Bellman, R., and Zadeh, L. A., "Decision Making in a Fuzzy Environment," Management Science, Vol. 17, No. 4, 1970, pp. 141-164.

[67] Weingartner, H. M., "Criteria for Programming Investment Project Selection," Journal of Industrial Economics, Vol. XV, No. 1, November 1966, pp. 65-76.

[68] Weingartner, H. M., Mathematical Programming and the Analysis of Capital Budgeting Problems, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1973.

APPENDIX A

A BRIEF LOOK AT MPSX

In this appendix, the IBM Mathematical Programming System Extended (MPSX), linear programming and mixed integer programming capabilities are summarized.

A.1 MPSX System

Mathematical Programming System Extended (MPSX) is composed of a set of procedures all operating under the direction of a user specified MPSX control program. Through the MPSX control program, the user specifies the sequence of steps to be executed in solving a mathermatical programming problem.

The user is able to augment MPSX with procedures written in the FORTRAN language through the use of the Read Communications Format (READCOMM) feature of MPSX. Through the use of FORTRAN CALL statements, the READCOMM subroutine is accessed. This subroutine acts as an interface between the MPSX control program and the FORTRAN procedures.

The user is capable of executing all of the MPSX capabilities through the use of the MPSX control programs and the READCOMM procedures.

A.2 Linear Programming Procedure

The MPSX strategy for solving a linear programming problem is the ordered execution of a series of the MPSX procedures. The user specifies the solution strategy to MPSX, via the MPSX control language.

The linear programming procedures of MPSX use the bounded variable/product form of the inverse/revised simplex. The simplex method is based upon the fact that if there are m constraints which are linearly independent, then there is a set of m columns (variables) which are also linearly independent. The right-hand side values can be expressed in terms of the m columns called a basis. The simplex

method employs these basic solutions by exchanging one column from the basis with one column not in the basis on each iteration, until a solution is realized that satisfies the feasibility criteria. This solution is termed a basic feasible solution.

The simplex method proceeds by examining the basic feasible solutions, to find one that satisfies the requirement that the objective function value be maximized or minimized.

A.3 Mixed Integer Programming Procedure

The Mixed Integer Programming capability of MPSX is an extension of the linear programming procedure of MPSX. It provides the user the capability to solve linear programming problems composed of both integer and continuous variables. This analysis is appropriate for the fuzzy linear programming approach to the capital budgeting problem. Since the solution must be of the form (0,1), the variables representing the investment possibilities must be integer values, while the value of the membership function is a continuous variable.

The MPSX mixed integer linear programming problem is performed in two stages. First, the problem is solved considering all integer variables as being continuous. The problem is solved by the linear programming capability of MPSX. The solution to this problem is termed the optimal continuous solution.

The second stage is to solve the problem for the optimal integer solution. The search for an integer solution starts from the optimal continuous solution and proceeds using the branch and bound technique. The search continues until the optimal integer solution is determined.

APPENDIX B

Fuzzy Linear Integer Programming Via MPSX/MIP Computer Code, Variable Definition Guide, User's Guide, and Sample Output

```
HEADLY M              .. HASP-II***..........START JOB 1718....NP229841......HEADLY M
HEADLY M              .. HASP-II***..........START JOB 1718....NP229841......HEADLY M


TIME: 15:33:21 DATE:  03/17/80                  H A S P   S Y S T E M   L O G


**NP229841 JOB 'U4771,T=0050,R=01500,S=280,      ','HEADLY M          '           AA
***FULLSKIPS
// EXEC SETTAF,TRAIN=TN,FORMS=16
***INCLUDE MGH01.$TODAY
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /SETTAF  / START 80077.1533
IEF374I STEP /SETTAF  / STOP  80077.1533 CPU   0MIN 00.03SEC MAIN   8K LCS    OK
// EXEC FHCG
//SYSIN DD *
//DATA.FT45F001 DD UNIT=SYSDA,DSN=&&NAMES,SPACE=(3200,(50,5)),
// DISP=(NEW,PASS),DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
//DATA.INPUT DD *
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /DATA    / START 80077.1533
IEF374I STEP /DATA    / STOP  80077.1533 CPU   0MIN 00.37SEC MAIN 280K LCS    OK
// EXEC LP
//SOURCE.INPUT DD *
***INCLUDE MGH01.$MISSY
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /SOURCE  / START 80077.1533
IEF374I STEP /SOURCE  / STOP  80077.1534 CPU   0MIN 00.70SEC MAIN 280K LCS    OK
//DATA.INPUT DD UNIT=SYSDA,DSN=&&NAMES,DISP=(OLD,DELETE)
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /DATA    / START 80077.1534
IEF374I STEP /DATA    / STOP  80077.1534 CPU   0MIN 00.47SEC MAIN 280K LCS    OK
IEF375I  JOB /NP229841/ START 80077.1533
IEF376I  JOB /NP229841/ STOP  80077.1534 CPU   0MIN 01.57SEC
```

```
C     ****************************************************************
C     ****************************************************************
C     **                                                            **
C     **        FUZZY INTEGER (0-1) LINEAR PROGRAMMING              **
C     **                                                            **
C     **        SOLUTION TECHNIQUE FOR THE MULTIOBJECTIVE           **
C     **                                                            **
C     **        CAPITAL BUDGETING PROBLEM VIA MPSX/MIP              **
C     **                                                            **
C     **        PREPARED BY CPT. MICHAEL G. HEADLY                  **
C     **                                                            **
C     **                                                            **
C     **                                                            **
C     **        ADDITIONAL INFORMATION CONCERNING FUZZY PROGRAMMING **
C     **                                                            **
C     **        MAY BE OBTAINED IN THE FOLLOWING REFERENCES         **
C     **                                                            **
C     **        1) ZIMMERMANN,H.J.,"DESCRIPTION AND OPTIMIZATION OF **
C     **                                                            **
C     **        FUZZY SYSTEMS",INTERNATIONAL JOURNAL OF GENERAL     **
C     **                                                            **
C     **        SYSTEMS,VOL.2,1976,PP.209-215.                      **
C     **                                                            **
C     **                                                            **
C     **        2) ZIMMERMANN,H.J.,"FUZZY PROGRAMMING WITH SEVERAL  **
C     **                                                            **
C     **        OBJECTIVE FUNCTIONS",INTERNATIONAL JOURNAL OF FUZZY **
C     **                                                            **
C     **        SETS AND SYSTEMS",VOL.1,1978,PP.45-55.              **
C     **                                                            **
C     **                                                            **
C     **        3) ZIMMERMANN,H.J.,"MEDIA SELECTION AND FUZZY LINEAR **
C     **                                                            **
C     **        PROGRAMMING",THE JOURNAL OF THE OPERATIONAL RESEARCH **
C     **                                                            **
```

```
C      **       SOCIETY",VOL.29,NO.11,1978,PP.1021-1084.                **
C      **                                                               **
C      **                                                               **
C      **       ADDITIONAL INFORMATION CONCERNING MPSX/MIP              **
C      **                                                               **
C      **       MAY BE OBTAINED IN THE FOLLOWING REFERENCES             **
C      **                                                               **
C      **                                                               **
C      **       MATHEMATICAL PROGRAMMING SYSTEM EXTENDED(MPSX),         **
C      **                                                               **
C      **       MIXED INTEGER PROGRAMMING(MIP) PROGRAM DESCRIPTION,     **
C      **                                                               **
C      **       PROGRAM REFERENCE MANUAL SH20-0908-1,AUGUST 1973.       **
C      **                                                               **
C      **                                                               **
C      *******************************************************************
C      *******************************************************************
C
C
C
C
C
C      *******************************************************************
C      **                                                               **
C      **      VARIABLE DEFINITION GUIDE                                **
C      **      *************************                                **
C      **                                                               **
C      **                                                               **
C      **      INPUT VARIABLES                                          **
C      **      ***************                                          **
C      **                                                               **
C      **      NVAR: NUMBER OF VARIABLES                                **
C      **                                                               **
C      **      RTYPE1: NUMBER OF RIGID CONSTRAINTS TYPE1(<)             **
C      **
```

```
C     **     X1(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **     ITH TYPE1 RIGID CONSTRAINT.                              **
C     **                                                              **
C     **     RHS1(I): RIGHT HAND SIDE OF THE ITH TYPE1 CONSTRAINT.    **
C     **                                                              **
C     **     KTYPE2: NUMBER OF RIGID CONSTRAINTS TYPE2(>)             **
C     **                                                              **
C     **     X2(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **     ITH TYPE2 RIGID CONSTRAINT.                              **
C     **                                                              **
C     **     RHS2(I): RIGHT HAND SIDE OF THE ITH TYPE2 CONSTRAINT.    **
C     **                                                              **
C     **     KTYPE3: NUMBER OF RIGID CONSTRAINTS TYPE3(=)             **
C     **                                                              **
C     **     X3(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **     ITH TYPE3 RIGID CONSTRAINT.                              **
C     **                                                              **
C     **     RHS3(I): RIGHT HAND SIDE OF THE ITH TYPE3 CONSTRAINT.    **
C     **                                                              **
C     **     ITYPE1: NUMBER OF TYPE1 FUZZY OBJECTIVE FUNCTIONS.       **
C     **                                                              **
C     **     X4(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **     ITH TYPE1 FUZZY OBJECTIVE FUNCTION.                      **
C     **                                                              **
C     **     ASP1(I): ASPIRATION LEVEL ITH TYPE1 FUZZY OBJECTIVE      **
C     **                                                              **
C     **     ITYPE2: NUMBER OF TYPE2 FUZZY OBJECTIVE FUNCTIONS.       **
C     **                                                              **
C     **     X5(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **     ITH TYPE2 FUZZY OBJECTIVE FUNCTION.                      **
C     **                                                              **
C     **     ASP2(I): ASPIRATION LEVEL ITH TYPE2 FUZZY OBJECTIVE      **
C     **                                                              **
C     **     ITYPE3: NUMBER OF TYPE3 FUZZY OBJECTIVE FUNCTIONS.       **
C     **                                                              **
```

```
C      **        X6(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C      **        ITH TYPE3 FUZZY OBJECTIVE FUNCTION.                      **
C      **                                                                 **
C      **        ASP3(I): ASPIRATION LEVEL ITH TYPE3 FUZZY OBJECTIVE      **
C      **                                                                 **
C      **        STINT(I): LOWEST ADMISSABLE VALUE OF THE ITH             **
C      **        TYPE3 FUZZY OBJECTIVE.                                   **
C      **                                                                 **
C      **        UTINT(I): GREATEST ADMISSABLE VALUE OF THE ITH           **
C      **        TYPE3 FUZZY OBJECTIVE.                                   **
C      **                                                                 **
C      **                                                                 **
C      **        INTERNAL VARIABLES                                       **
C      **        ******************                                       **
C      **                                                                 **
C      **        ROW(I,J): COEFFICENT OF THE JTH VARIABLE IN THE ITH      **
C      **        CONSTRAINT/OBJECTIVE FORMULATED FOR INPUT INTO MPSX.     **
C      **                                                                 **
C      **        RHS(I): RIGHT-HAND SIDE VALUE OF THE ITH CONSTRAINT/     **
C      **        OBJECTIVE FORMULATED FOR INPUT INTO MPSX.                **
C      **                                                                 **
C      **                                                                 **
C      **        MPSX INPUT COMMANDS                                      **
C      **        *******************                                      **
C      **                                                                 **
C      **        INTORG AND INTEND: MPSX COMMANDS WHICH IDENTIFY THOSE    **
C      **        VARIABLES WHICH MUST BE INTEGER VALUED.                  **
C      **                                                                 **
C      **        UPBOUND: MPSX COMMAND WHICH ESTABLISHES THE              **
C      **        UPPER BOUND OF THOSE VARIABLES THAT ARE BOUNDED.         **
C      **                                                                 **
C      **                                                                 **
C      *******************************************************************
C
```

```
C     ****************************************************************
C     **                                                            **
C     **      USER'S GUIDE                                          **
C     **      ************                                          **
C     **                                                            **
C     **      CARD SET                   DATA INPUT FORMAT          **
C     **      ********                   *****************          **
C     **                                                            **
C     **      CARD SET # 1                       9F8.4              **
C     **      NUMBER OF CARDS IN SET: 1                             **
C     **                                                            **
C     **          NVAR                                              **
C     **          KTYPE1                                            **
C     **          KTYPE2                                            **
C     **          KTYPE3                                            **
C     **          ITYPE1                                            **
C     **          ITYPE2                                            **
C     **          ITYPE3                                            **
C     **                                                            **
C     **      CARD SET # 2                       9F8.4              **
C     **      NUMBER OF CARDS IN SET : KTYPE1+1                     **
C     **      RIGID CONSTRAINT TYPE # 1                             **
C     **                                                            **
C     **          ((X1(J,I),I=1,NVAR),J=1,KTYPE1)                   **
C     **          (RHS1(I),I=1,KTYPE1)                              **
C     **                                                            **
C     **      CARD SET # 3                       9F8.4              **
C     **      NUMBER OF CARDS IN SET : KTYPE2+1                     **
C     **      RIGID CONSTRAINT TYPE # 2                             **
C     **          ((X2(J,I),I=1,NVAR),J=1,KTYPE2)                   **
C     **          (RHS2(I),I=1,KTYPE2)                              **
C     **                                                            **
C     **      CARD SET # 4                       9F8.4              **
C     **      NUMBER OF CARDS IN SET : KTYPE3+1                     **
C     **      RIGID CONSTRAINT TYPE # 3                             **
C     **          ((X3(J,I),I=1,NVAR),J=1,KTYPE3)                   **
C     **                                                            **
```

```
C      **          (RES3(I),I=1,KTYPE3)                                       **
C      **        CARD SET # 5                    9F8.4                        **
C      **        NUMBER OF CARDS IN SET : ITYPE1+1                            **
C      **        FUZZY CONSTRAINT TYPE # 1                                    **
C      **          ((X4(J,I),I=1,NVAR),J=1,ITYPE1)                            **
C      **          (ASP1(I),TINT1(I),I=1,ITYPE1)                              **
C      **                                                                     **
C      **        CARD SET # 6                    9F8.4                        **
C      **        NUMBER OF CARDS IN SET : ITYPE2+1                            **
C      **        FUZZY CONSTRAINT TYPE # 2                                    **
C      **          ((X5(J,I),I=1,NVAR),J=1,ITYPE2)                            **
C      **          (ASP2(I),TINT2(I),I=1,ITYPE2)                              **
C      **                                                                     **
C      **        CARD SET # 7                    9F8.4                        **
C      **        NUMBER OF CARDS IN SET : ITYPE3+1                            **
C      **        FUZZY CONSTRAINT TYPE # 3                                    **
C      **          ((X6(J,I),I=1,NVAR),J=1,ITYPE3)                            **
C      **          (ASP3(I),STINT(I),UTINT(I),I=1,ITYPE3)                     **
C      **                                                                     **
C      **                                                                     **
C      *************************************************************************
C
C
C
C
```

```
C
C     *******************************************
C     **    DATA SET 45 IS THE TEMPORARY       **
C     **    DATA SET THAT IS USED AS INPUT     **
C     **    FOR THE MPSX/MIP ROUTINE.          **
C     *******************************************
      DIMENSION X1(25,25),X2(25,25),X3(25,25),X4(25,25),
     1X5(25,25),X6(25,25),RHS(25),ROW(25,25),RHS1(25),RHS2(25),
     1RHS3(25),ASP1(25),TINT1(25),ASP2(25),TINT2(25),
     1ASP3(25),STINT(25),UTINT(25)
      READ 1,NVAR,KTYPE1,KTYPE2,KTYPE3,ITYPE1,ITYPE2,ITYPE3
      NFUZZY=ITYPE1+ITYPE2+ITYPE3*2
      NRES=NFUZZY+KTYPE1+KTYPE2+KTYPE3
      M=0
C     ***
C     ***
C     ***  RIGID CONSTRAINT TYPE 1 (<).
C     ***
C     ***
      IF(KTYPE1.EQ.0) GC TC 30
      DO 50 IC=1,KTYPE1
   50 READ 2,(X1(IC,I),I=1,NVAR)
      READ 2,(RHS1(I),I=1,KTYPE1)
      DO 100 I=1,KTYPE1
      M=M+1
      DO 200 J=1,NVAR
      ROW(M,J)=X1(I,J)
      RHS(M)=RHS1(I)
  200 CONTINUE
  100 CONTINUE
   30 M=KTYPE1
C     ***
C     ***
C     ***  RIGID CONSTRAINT TYPE 2 (>)
```

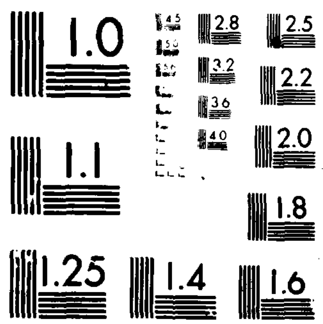
```
C       ***
C       ***
        IF(KTYPE2.EQ.0) GO TO 31
        DO 51 IA=1,KTYPE2
   51   READ 2,(X2(IA,I),I=1,NVAR)
        READ 2,(RHS2(I),I=1,KTYPE2)
        DO 300 I=1,KTYPE2
        M=M+1
        DO 400 J=1,NVAR
        ROW(M,J)=X2(I,J)
        RHS(M)=RHS2(I)
  400   CONTINUE
  300   CONTINUE
C       ***
C       ***
C       ***  RIGID CONSTRAINT TYPE 3 (=)
C       ***
C       ***
   31   M=KTYPE1+KTYPE2
        IF(KTYPE3.EQ.0) GO TO 32
        DO 52 IB=1,KTYPE3
   52   READ 2,(X3(IB,I),I=1,NVAR)
        READ 2,(RHS3(I),I=1,KTYPE3)
        DO 500 I=1,KTYPE3
        M=M+1
        DO 600 J=1,NVAR
        ROW(M,J)=X3(I,J)
        RHS(M)=RHS3(I)
  600   CONTINUE
  500   CONTINUE
C       ***
C       ***
C       ***  FUZZY CONSTRAINT TYPE 1
C       ***
C       ***
```

AD-A083 792   ARMY MILITARY PERSONNEL CENTER ALEXANDRIA VA   F/G 5/3
AN ANALYSIS OF THE MULTIPLE OBJECTIVE CAPITAL BUDGETING PROBLEM--ETC(U)
MAY 80   M G HEADLY

UNCLASSIFIED   NL

2 OF 2
AD
A083792

END
DATE
FILMED
6-80
DTIC


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

```
 32   M=KTYPE1+KTYPE2+KTYPE3
      IF(ITYPE1.EQ.0) GO TO 33
      DO 60 ID=1,ITYPE1
 60   READ 2,(X4(ID,I),I=1,NVAR)
      READ 2,(ASP1(I),TINT1(I),I=1,ITYPE1)
      DO 700 I=1,ITYPE1
      M=M+1
      DO 800 J=1,NVAR
      ROW(M,J)=-X4(I,J)/TINT1(I)
      RHS(M)=(TINT1(I)-ASP1(I))/TINT1(I)
800   CONTINUE
700   CONTINUE
C     ***
C     ***
C     ***   FUZZY CONSTRAINT TYPE 2
C     ***
C     ***
 33   M=KTYPE1+KTYPE2+KTYPE3+ITYPE1
      IF(ITYPE2.EQ.0) GO TO 34
      DO 53 IF=1,ITYPE2
 53   READ 2,(X5(IF,I),I=1,NVAR)
      READ 2,(ASP2(I),TINT2(I),I=1,ITYPE2)
      DO 900 I=1,ITYPE2
      M=M+1
      DO 1000 J=1,NVAR
      ROW(M,J)=X5(I,J)/TINT2(I)
      RHS(M)=(TINT2(I)+ASP2(I))/TINT2(I)
1000  CONTINUE
900   CONTINUE
C     ***
C     ***
C     ***   FUZZY CONSTRAINT TYPE 3
C     ***
C     ***
 34   M=KTYPE1+KTYPE2+KTYPE3+ITYPE1+ITYPE2
```

```
      IF(ITYPE3.EQ.0) GO TO 35
      DO 54 IG=1,ITYPE3
   54 READ 2,(X6(IG,I),I=1,NVAR)
      READ 2,(ASP3(I),STINT(I),UTINT(I),I=1,ITYPE3)
      DO 1100 I=1,ITYPE3
      M=M+1
      MCOUNT=M
      DO 1200 J=1,NVAR
      ROW(M,J)=-X6(I,J)/STINT(I)
      RHS(M)=-(ASP3(I)-STINT(I))/STINT(I)
      M=M+1
      ROW(M,J)=X6(I,J)/UTINT(I)
      RHS(M)=(UTINT(I)+ASP3(I))/UTINT(I)
      IF(J.EQ.NVAR) GO TO 1200
      M=M-1
 1200 CONTINUE
      KKJ=MCOUNT
      KKJ=KKJ+1
      KKJ=0
 1100 CONTINUE
C     ***
C     ***  THE REMAINDER OF THE PROGRAM WRITES
C     ***
C     ***  THE TRANSFORMED FUNSTIONS INTO
C     ***
C     *** THE TEMPORARY DATA SET FOR INPUT INTO
C     ***
C     ***  THE MPSX/MIP ROUTINE.
C     ***
   35 WRITE(45,8)
    8 FORMAT('NAME',10X,'MAX')
      WRITE(45,9)
    9 FORMAT('ROWS')
      WRITE(45,10)
   10 FORMAT(2X,'N',1X,'OBJ')
      M=0
```

```
      IF(KTYPE1.EQ.0) GO TO 36
      DO 2000 I=1,KTYPE1
      M=I
      M=M+100000
      WRITE(45,11) M
 11   FORMAT(2X,'L',1X,I6,T5,'ROW')
      M=M-100000
 2000 CONTINUE
 36   M=KTYPE1
      IF(KTYPE2.EQ.0) GO TO 37
      DO 3000 I=1,KTYPE2
      M=M+1
      M=M+100000
      WRITE(45,12) M
 12   FORMAT(2X,'G',1X,I6,T5,'ROW')
      M=M-100000
 3000 CONTINUE
 37   M=KTYPE1+KTYPE2
      IF(KTYPE3.EQ.0) GO TO 38
      DO 4000 I=1,KTYPE3
      M=M+1
      M=M+100000
      WRITE(45,13) M
 13   FORMAT(2X,'E',1X,I6,T5,'ROW')
      M=M-100000
 4000 CONTINUE
 38   M=KTYPE1+KTYPE2+KTYPE3
      IF(ITYPE1.EQ.0) GO TO 39
      DO 5000 I=1,ITYPE1
      M=M+1
      M=M+100000
      WRITE(45,11) M
      M=M-100000
 5000 CONTINUE
 39   M=KTYPE1+KTYPE2+KTYPE3+ITYPE1
```

```
      IF(ITYPE2.EQ.0) GO TO 40
      DO 6000 I=1,ITYPE2
      M=M+1
      M=M+100000
      WRITE(45,11) M
      M=M-100000
 6000 CONTINUE
   40 M=KTYPE1+KTYPE2+KTYPE3+ITYPE1+ITYPE2
      IF(ITYPE3.EQ.0)GO TO 41
      K=M+ITYPE3*2
   42 M=M+1
      M=M+100000
      WRITE(45,11) M
      M=M+1
      WRITE(45,11) M
      M=M-100000
      IF(M.NE.K) GO TO 42
   41 WRITE(45,14)
   14 FORMAT('COLUMNS')
      WRITE(45,15)
   15 FORMAT(4X,'LAMBDA',4X,'OBJ',9X,'1.0')
      L=ITYPE1+ITYPE2+ITYPE3*2
      M=KTYPE1+KTYPE2+KTYPE3
      DO 8000 I=1,L
      M=M+1
      M=M+100000
      WRITE(45,16) M
   16 FORMAT(4X,'LAMBDA',4X,I6,T15,'ROW',7X,'1.0')
      M=M-100000
 8000 CONTINUE
      WRITE(45,45)
   45 FORMAT(4X,'INTORG',4X,'''MARKER''',17X,'''INTORG''')
      M=0
      L=0
      L=KTYPE1+KTYPE2+KTYPE3+ITYPE1+ITYPE2+ITYPE3*2
```

```
      DO 8050 J=1,NVAR
      DO 9000 I=1,I
      M=M+1
      IF(M.EQ.L+1) M=1
      N=J
      K=M
      N=N+1000
      K=K+100000
      WRITE(45,17) N,K,ROW(M,J)
17    FORMAT(4X,I4,T5,'X',9X,I6,T15,'ROW',7X,F8.4)
      N=N-1000
      K=K-100000
9000  CONTINUE
8050  CONTINUE
      WRITE(45,46)
46    FORMAT(4X,'INTEND',4X,'''MARKER''',17X,'''INTEND''')
      WRITE(45,18)
18    FORMAT('RHS')
      L=KTYPE1+KTYPE2+KTYPE3+ITYPE1+ITYPE2+ITYPE3*2
      DO 9050 I=1,L
      N=I
      N=N+100000
      WRITE(45,19) N,RHS(I)
19    FORMAT(4X,'RHS',7X,I6,T15,'ROW',7X,F8.4)
      N=N-100000
9050  CONTINUE
      WRITE(45,47)
47    FORMAT('BOUNDS')
      WRITE(45,49)
49    FORMAT(1X,'UP',1X,'UPBOUND',3X,'LAMBDA',4X,'1.0')
      DO 9075 J=1,NVAR
      N=J
      N=N+1000
      WRITE(45,48) N
48    FORMAT(1X,'UP',1X,'UPBOUND',3X,I4,T15,'X',10X,'1.0')
      N=N-1000
```

```
 9075 CONTINUE
      WRITE(45,20)
   20 FORMAT('ENDATA')
    1 FORMAT(7I5)
    2 FORMAT(9F8.4)
      STOP
      END

ECUTED=     1775

   OBJECT CODE=    15048 BYTES,ARRAY AREA=    18600 BYTES,TOTAL AREA AVAILABLE=   178176

     NUMBER OF ERRORS=        0, NUMBER OF WARNINGS=        0, NUMBER OF EXTENSIONS=

.=     0.22 SEC,EXECUTION TIME=     0.08 SEC,
```

```
..MPSX-V1M7..  CONTROL PROGRAM COMPILER.  MPSX  RELEASE 1  MOD LEVEL 7

     0001                PROGRAM
     0002                INITIALZ
     0096                MOVE(XDATA,'MAX')
     0097                MOVE(XPBNAME,'BUDGET')
     0098                CONVERT
     0099                SETUP('BOUND','UPBOUND','MAX')
     0100                MOVE(XOBJ,'CBJ')
     0101                MOVE(XRHS,'RHS')
     0102                OPTIMIZE
     0411                SOLUTION
     0412                SAVE('NAME','CPTC')
     0413                INIMIX
     0451                MIXSTART('CCST')
     0452                XMXDROP=0.0
     0453                CT=0
     0454                MVADR(XDOPRINT,INT)
     0455                MIXFLOW
     0456      STOP      MIXSAVE('NAME','TREE1')
     0457                MIXSTATS('NODES')
     0458                EXIT
     0459      INT       SOLUTION
     0460                XMXDROP=0.0
     0461                CT=CT+1
     0462                IF(CT.EQ.5,STOP)
     0463                CONTINUE
     0464      CT        DC(0)
     0465                PEND
```

```
..MPSX-V1M7..  EXECUTOR.  MPSX  RELEASE 1  MOD LEVEL 7
CONVERT MAX       TO BUDGET
TIME =    0.00
1- ROWS SECTION.

   0 MINOR ERROR(S) -   0 MAJOR ERROR(S).
2- COLUMNS SECTION.

   0 MINOR ERROR(S) -   0 MAJOR ERROR(S).
3- RHS'S SECTION.
  RHS
   0 MINOR ERROR(S) -   0 MAJOR ERROR(S).
5- BOUNDS SECTION.
  UPBOUND
   0 MINOR ERROR(S) -   0 MAJOR ERROR(S).
 PROBLEM STATISTICS
    12 LP ROWS,      23 VARIABLES,      92 LP ELEMENTS, DENSITY =  33.33
                                10 INTEGER VARIABLES
 THESE STATISTICS CONTAIN ONE SLACK VARIABLE FOR EACH ROW
     0 MINOR ERRORS,     0 MAJOR ERRORS.
```

..MPSX-V147.. EXECUTOR. MPSX RELEASE 1 MOD LEVEL 7

SETUP BUDGET

TIME = 0.00

BOUND = UPBOUND
MAX
SCALE

MATRIX1 ASSIGNED TO MATRIX1
MATRIX2 ASSIGNED TO MATRIX2

ETA1 ASSIGNED TO ETA1

SCRATCH1 ASSIGNED TO SCRATCH1
SCRATCH2 ASSIGNED TO SCRATCH2

MIXWORK ASSIGNED TO MIXWORK

MAXIMUM PRICING NOT REQUIRED - MAXIMUM POSSIBLE 7

NO CYCLING

POOLS	NUMBER	SIZE	CORE
H.REG-BITS MAP			72
INTEGER TABLES			1568
BOUND VECTOR			120
WORK REGIONS	9	120	1080
MATRIX BUFFERS	3	600	1800
ETA BUFFERS	3	2304	6912

	TOTAL	NORMAL	.FREE.	FIXED	BOUNDED
ROWS (LOG.VAR.)	12	6	1	5	0
COLUMNS (STR.VAR.)	11	0	0	0	11

```
      10 COLUMNS ARE INTEGER

       92 ELEMENTS - DENSITY =   33.33 -      3 MATRIX RECORDS (WITHOUT RHS'S)


WRITE

TIME =    0.00

    OPTIMIZE SYSTEM MACRO CALLED

CRASH          TIME   0.00 MINS.
        INFEASIBILITIES       8  AT START
                              8  AFTER PASS A


 INVERT CALLED      TIME    0.00   CURRENT INVERSE ---- ETA-VECTORS .....1    ELEMENTS .....1
  BASIS ---- NO.OF ROWS ....12     LOGICALS .......12   STRUCTURALS .....0    ELEMENTS ....12

  INVERSE -- NUCLEUS ........0     TRANSFORMED .....0   ETA-VECTORS .....1    ELEMENTS .....1


                              0  AFTER PASS B
                          FEASIBLE


NEGDI

TIME =     0.00  MINS.

PRIMAL       OBJ = OBJ         RHS = RHS

TIME =     0.00  MINS.        PRICING    7
 SCALE =    1.00000-
```

```
   ITER    NUMBER  VECTOR  VECTOR  REDUCED    FUNCTION
  NUMBER  NONOPT    OUT      IN     COST        VALUE
M    10       1       11      13   1.00000-    .13340
M    11       1       16      14    .33182-    .13340
 INVERT DEMANDED AFTER  10 MAJOR/  10 MINOR ITERATIONS - CLOCK CONTROL

 INVERT CALLED      TIME   0.00   CURRENT INVERSE ---- ETA-VECTORS .....7   ELEMENTS ....44
  BASIS ---- NO.OF ROWS ....12    LOGICALS ........7   STRUCTURALS .....5   ELEMENTS ....42
  INVERSE -- NUCLEUS ........0    TRANSFORMED .....0   ETA-VECTORS .....6   ELEMENTS ....36

NEGOJ

TIME =    0.00  MINS.

PRIMAL       OBJ = OBJ          RHS = RHS

TIME =    0.00  MINS.        PRICING    7
 SCALE =    1.00000-

   ITER    NUMBER  VECTOR  VECTOR  REDUCED    FUNCTION
  NUMBER  NONOPT    OUT      IN     COST        VALUE
M    12       1        2      17    .46110-    .50945
M    13       1        6      18    .40714-    .50945
M    14       1       22      23   2.50491-    .50945
M    15       1        8      15    .61259     .58071
M    16       1       20      22    .54614-    .58071
 INVERT DEMANDED AFTER   5 MAJOR/   5 MINOR ITERATIONS - CLOCK CONTROL

 INVERT CALLED      TIME   0.00   CURRENT INVERSE ---- ETA-VECTORS ....11   ELEMENTS ....85
  BASIS ---- NO.OF ROWS ....12    LOGICALS ........4   STRUCTURALS .....8   ELEMENTS ....62
  INVERSE -- NUCLEUS ........6    TRANSFORMED .....2   ETA-VECTORS ....14   ELEMENTS ....65

NEGOJ

TIME =    0.00  MINS.

PRIMAL       OBJ = OBJ          RHS = RHS
```

```
..MPSX-V1M7..  EXECUTOR.  MPSX  RELEASE 1  MOD LEVEL 7

TIME =     0.00  MINS.        PRICING    7
 SCALE =    1.00000-
 OPTIMAL SOLUTION
..MPSX-V1M7..  EXECUTOR.  MPSX  RELEASE 1  MOD LEVEL 7

SOLUTION   (OPTIMAL)

TIME =    0.00 MINS. ITERATION NUMBER =     16

          ...NAME...     ...ACTIVITY...       DEFINED AS

          FUNCTIONAL           .58071         OBJ
          RESTRAINTS                          RHS
          BOUNDS....                          UPBOUND
1M7..  EXECUTOR.  MPSX  RELEASE 1  MOD LEVEL 7

1 - ROWS
```

...ROW..	AT	...ACTIVITY...	SLACK ACTIVITY	..LOWER LIMIT.	..UPPER LIMIT.	.DUAL ACTIVITY
OBJ	BS	.58071	.58071-	NONE	NONE	1.00000
ROW001	LL	.16250-	.	.16250-	NONE	9.44695
ROW002	EQ	1.00000	.	1.00000	1.00000	2.03095
ROW003	EQ	1.00000	.	1.00000	1.00000	1.96510
ROW004	EQ	1.00000	.	1.00000	1.00000	.80932
ROW005	EQ	1.00000	.	1.00000	1.00000	.09136-
ROW006	EQ	.	.	.	.	.29727-
ROW007	UL	4.00000	.	NONE	4.00000	.95977-
ROW008	BS	4.53436-	.53436	NONE	4.00000-	.
ROW009	BS	10.81078	.18922	NONE	11.00000	.
ROW010	UL	2.33330-	.	NONE	2.33330-	.04023-
ROW011	BS	9.32265	1.67735	NONE	11.00000	.

```
..MPSX-V1M7..  EXECUTOR.  MPSX  RELEASE 1  MOD LEVEL 7

SECTION 2 - COLUMNS

 NUMBER  .COLUMN.  AT  ...ACTIVITY...  ..INPUT COST..  ..LOWER LIMIT.  ..UPPER LIMIT.

     13  LAMBDA    BS         .58071         1.00000          .             1.00000
     14  X001      BS         .06172          .               .             1.00000
     15  X002      BS         .93828          .               .             1.00000
     16  X003      LL         .               .               .             1.00000
     17  X004      BS         .32810          .               .             1.00000
     18  X005      BS         .06172          .               .             1.00000
     19  X006      BS         .61018          .               .             1.00000
     20  X007      LL         .               .               .             1.00000
     21  X008      UL        1.00000          .               .             1.00000
     22  X009      BS         .               .               .             1.00000
     23  X010      BS         .               .               .             1.00000


..MPSX-V1M7..  EXECUTOR.  MPSX  RELEASE 1  MOD LEVEL 7

MIXSTART - TIME =    0.00

   COST

MIXFLOW - TIME =    0.00
```

```
-----------------------------------------------------------------------------------------------
I NODES POSTPONED IF ESTIMATION BEYOND .10000E+76-                                            I
I NODES DROPPED    IF FUNCTIONAL BEYOND .                                                     I
I REACH    NODE          1 - FUNCTIONAL =           .5807  ESTIMATION =  .58071     ITER ......16 I
I FIRST  BRANCH - VARIABLE     17 =    .3281     FORCED TOWARDS UPPER ACTIVITIES              I
-----------------------------------------------------------------------------------------------

    ITER    NUMBER   VECTOR   VECTOR   REDUCED    FUNCTION   PARAM
   NUMBER   NONOPT    OUT       IN      COST       VALUE     VALUE
M    17       0        17       20     .21563     .58071    .32810
M    18       0        22       21     .13532-    .58071    .32810
M    19       0        18       11     .13981     .56528    .36271
M    20       0        19       16     .09256     .37256    .58091
 BRANCH DROPPED

-----------------------------------------------------------------------------------------------
I RESTORE NODE          1 - FUNCTIONAL =           .5807  ESTIMATION =  .58071                I
I SECOND BRANCH - VARIABLE     17 =    .3281     FORCED TOWARDS LOWER ACTIVITIES              I
-----------------------------------------------------------------------------------------------
M    21       0        17       21     .01927-    .58071    .67190
M    22       0        22       11     .03390     .56633    .92741

-----------------------------------------------------------------------------------------------
I REACH    NODE          2 - FUNCTIONAL =           .5506  ESTIMATION =  .55058     ITER ......22 I
I BEST POSSIBLE SOLUTION IS NOW              .55058                                           I
I FIRST  BRANCH - VARIABLE     14 =    .6967     FORCED TOWARDS LOWER ACTIVITIES              I
-----------------------------------------------------------------------------------------------
M    23       0        14        2     .24577     .55058    .30328
M    24       0        11       16     .50121     .53262    .59577

-----------------------------------------------------------------------------------------------
I REACH    NODE          3 - FUNCTIONAL =           .3190  ESTIMATION =  .31897     ITER ......24 I
I RESTORE NODE          2 - FUNCTIONAL =           .5506  ESTIMATION =  .55058                I
I SECOND BRANCH - VARIABLE     14 =    .6967     FORCED TOWARDS UPPER ACTIVITIES              I
-----------------------------------------------------------------------------------------------
M    25       0        14       22     .05501-    .55058    .69672
M    26       0        12        8    2.22737     .54413    .82795
 BRANCH DROPPED
```

```
-------------------------------------------------------------------------------------------
I BEST POSSIBLE SOLUTION IS NOW          .31897                                            I
I RESTORE NODE        3 - FUNCTIONAL =             .3190  ESTIMATION =  .31897             I
I FIRST  BRANCH - VARIABLE     15 =    .6591    FORCED TOWARDS LOWER ACTIVITIES            I
-------------------------------------------------------------------------------------------
M    27        0        15        11     .39711     .31897     .34091
 BRANCH DROPPED


-------------------------------------------------------------------------------------------
I RESTORE NODE        3 - FUNCTIONAL =             .3190  ESTIMATION =  .31897             I
I SECOND BRANCH - VARIABLE     15 =    .6591    FORCED TOWARDS UPPER ACTIVITIES            I
-------------------------------------------------------------------------------------------
M    28        0        15        22     .00265-    .31897     .65909
M    29        0        21        20     .09723     .31699     .82233
M    30        0        22        21     .03283-    .31699     .82233

-------------------------------------------------------------------------------------------
I REACH   NODE        4 - FUNCTIONAL =             .2841  ESTIMATION =  .28406      ITER ......30 I
I BEST POSSIBLE SOLUTION IS NOW          .28406                                            I
I FIRST  BRANCH - VARIABLE     19 =    .8326    FORCED TOWARDS LOWER ACTIVITIES            I
-------------------------------------------------------------------------------------------

INVERT CALLED      TIME    0.00   CURRENT INVERSE ---- ETA-VECTORS ....21   ELEMENTS ...138   RECC
 BASIS ---- NO.OF ROWS ....12     LOGICALS ........5   STRUCTURALS .....7   ELEMENTS ....56
 INVERSE -- NUCLEUS ........5     TRANSFORMED .....1   ETA-VECTORS ....13   ELEMENTS ....58   RECC

MIXFLOW - TIME =   0.00

   ITER    NUMBER   VECTOR   VECTOR   REDUCED    FUNCTION    PARAM
  NUMBER   NONOPT    OUT       IN       COST       VALUE      VALUE
M    31        0       19        22     .03788     .28406     .16740
M    32        0        9        11     .40580     .26758     .25701
 BRANCH DROPPED
```

```
-----------------------------------------------------------------------------------------------
I RESTORE NODE        4 - FUNCTIONAL =           .2841  ESTIMATION =  .28406                  I
I SECOND BRANCH - VARIABLE     19 =    .8326     FORCED TOWARDS UPPER ACTIVITIES               I
-----------------------------------------------------------------------------------------------
M    33        0        19         8  1.18997    .28406     .83260

-----------------------------------------------------------------------------------------------
I REACH   NODE        5 - FUNCTIONAL =           .1334  ESTIMATION =  .13340       ITER ......33 I
I  - - - - - - - -  INTEGER SOLUTION                                                          I
-----------------------------------------------------------------------------------------------
```

```
..MPSX-V1M7..  EXECUTOR.  MPSX  RELEASE 1  MOD LEVEL 7

SOLUTION

TIME =   0.00 MINS. ITERATION NUMBER =     33

          ...NAME...     ...ACTIVITY...       DEFINED AS

          FUNCTIONAL            .13340        OBJ
          RESTRAINTS                          RHS
          BOUNDS....                          UIBOUND
```

- ROWS

...ROW..	AT	...ACTIVITY...	SLACK ACTIVITY	..LOWER LIMIT.	..UPPER LIMIT.	.DUAL ACTIVITY
OBJ	BS	.13340	.13340-	NONE	NONE	1.00000
ROW001	BS	.14380-	.01870-	.16250-	NONE	.
ROW002	EQ	1.00000	.	1.00000	1.00000	.33330-
ROW003	EQ	1.00000	.	1.00000	1.00000	1.80000-
ROW004	EQ	1.00000	.	1.00000	1.00000	.40000-
ROW005	EQ	1.00000	.	1.00000	1.00000	.96670-
ROW006	EQ	.	.	.	.	.13330
ROW007	BS	3.71800	.28200	NONE	4.00000	.
ROW008	BS	4.56660-	.56660	NONE	4.00000-	.
ROW009	BS	9.53340	1.46660	NONE	11.00000	.
ROW010	UL	2.33330-	.	NONE	2.33330-	1.00000-
ROW011	BS	7.53340	3.46660	NONE	11.00000	.

..MPSX-V1M7.. EXECUTOR. MPSX RELEASE 1 MOD LEVEL 7

SECTION 2 - COLUMNS

NUMBER	.COLUMN.	AT	...ACTIVITY...	..INPUT COST..	..LOWER LIMIT.	..UPPER LIMIT.
13	LAMBDA	BS	.13340	1.00000	.	1.00000
14	X001	IV	.	.	.	1.00000
15	X002	IV	1.00000	.	.	1.00000
16	X003	IV	.	.	.	1.00000
17	X004	IV	.	.	.	1.00000
18	X005	IV	.	.	.	1.00000
19	X006	IV	1.00000	.	.	1.00000
20	X007	IV	.	.	.	1.00000
21	X008	IV	1.00000	.	.	1.00000
22	X009	IV	.	.	.	1.00000
23	X010	IV	.	.	.	1.00000

..MPSX-V1M7.. EXECUTOR. MPSX RELEASE 1 MOD LEVEL 7

MIXFLOW - TIME = 0.00

```
------------------------------------------------------------------------------
I BEST POSSIBLE SOLUTION IS NOW        .13340                                 I
I RESTORE MODE        5 - FUNCTIONAL =        .1334  ESTIMATION =  .13340     I
I   O P T I M A L   INTEGER SOLUTION                                          I
------------------------------------------------------------------------------
```

```
..MPSX-V1M7..   EXECUTOR.   MPSX   RELEASE 1   MOD LEVEL 7

                                        WAITING AND INTEGER NODES

I-----------------I-------------I
I                 I             I
I      NODE       I          5  I
I                 I             I
I-----------------I-------------I
I                 I             I
I   FUNCTIONAL    I      .1334  I
I                 I             I
I-----------------I-------------I
I                 I             I
I   ESTIMATION    I   INTEGER   I
I                 I             I
I-----------------I-------------I
I                 I             I
I     14=  X001   I     .       I
I     15=  X002   I    1.0000   I
I     16=  X003   I     .       I
I     17=  X004   I     .       I
I     18=  X005   I     .       I
I     19=  X006   I    1.0000   I
I     20=  X007   I     .       I
I     21=  X008   I    1.0000   I
I     22=  X009   I     .       I
I     23=  X010   I     .       I
I                 I             I
I-----------------I-------------I

ACCOUNT: H4771                 MAXIMUM TIME (SEC):    50  NET CPU (SEC):        2
DATE: 03/17/80 IDENT:          ACTUAL TIME, INCLUDING 2.0 SEC SYSTEM TIME:     39
USER: HEADLY M                 LINES PRINTED:        805  CARDS PUNCHED:        0
DESTINATION: AA                MAXIMUM RECORDS:     1500  TOTAL RECORDS:      805
OS-21.8 HASP-2.PSG 370/3033    CARDS READ:           521                    *****
```

APPENDIX C

Fuzzy Linear Integer Programming Via an Exchange Heuristic, Variable Definition Guide, User's Guide and Sample Output

```
HEADLY M            .. HASP-II***.........START JOB 1305....NP129486......HEADLY M
HEADLY M            .. HASP-II***.........START JOB 1305....NP129486......HEADLY M
HEADLY M            .. HASP-II***.........START JOB 1305....NP129486......HEADLY M


TIME: 15:12:48 DATE:  03/17/80                H A S P   S Y S T E M   L O G

**NP129486 JOB 'U4771,T=0025,R=02500,S=280,      ','HEADLY M          '         AA
***FULLSKIPS                                                            00000050
// EXEC SETTAF,TRAIN=TN,FORMS=16                                         00000100
***INCLUDE MGH01.$ONE                                                   00000150
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /SETTAF  / START 80077.1512
IEF374I STEP /SETTAF  / STOP  80077.1512 CPU   0MIN 00.04SEC MAIN   8K LCS    0K
// EXEC FWCG
//SYSIN DD *
//DATA.FT80F001 DD UNIT=SYSDA,DSN=&&EXDATA,SPACE=(3200,(50,5)),
// DISP=(NEW,PASS),DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
//DATA.INPUT DD *
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /DATA    / START 80077.1512
IEF374I STEP /DATA    / STOP  80077.1512 CPU   0MIN 00.36SEC MAIN 280K LCS    0K
// EXEC FWCG
//SYSIN DD *
***INCLUDE MGH01.$TWO                                                   00000200
***INCLUDE MGH01.$THREE                                                 00000250
//DATA.FT80F001 DD UNIT=SYSDA,DSN=&&EXDATA,DISP=(OLD,DELETE)
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /DATA    / START 80077.1512
IEF374I STEP /DATA    / STOP  80077.1512 CPU   0MIN 01.34SEC MAIN 280K LCS    0K
IEF375I  JOB /NP129486/ START 80077.1512
IEF376I  JOB /NP129486/ STOP  80077.1512 CPU   0MIN 01.74SEC
```

```
C     ********************************************************************
C     ********************************************************************
C     **                                                                **
C     **        FUZZY INTEGER (0-1) LINEAR PROGRAMMING                  **
C     **                                                                **
C     **        SOLUTION TECHNIQUE FOR THE MULTIOBJECTIVE               **
C     **                                                                **
C     **        CAPITAL BUDGETING PROBLEM VIA A MODIFIED VERSION        **
C     **                                                                **
C     **        OF C.C. PETERSEN'S EXCHANGE HEURISTIC ALGORITHM.        **
C     **                                                                **
C     **        PREPARED BY CPT. MICHAEL G. HEADLY                      **
C     **                                                                **
C     **                                                                **
C     **                                                                **
C     **        ADDITIONAL INFORMATION CONCERNING FUZZY PROGRAMMING     **
C     **                                                                **
C     **        MAY BE OBTAINED IN THE FOLLOWING REFERENCES             **
C     **                                                                **
C     **        1) ZIMMERMANN,H.J.,"DESCRIPTION AND OPTIMIZATION OF     **
C     **                                                                **
C     **        FUZZY SYSTEMS",INTERNATIONAL JOURNAL OF GENERAL         **
C     **                                                                **
C     **        SYSTEMS,VOL.2,1976,PP.209-215.                          **
C     **                                                                **
C     **                                                                **
C     **        2) ZIMMERMANN,H.J.,"FUZZY PROGRAMMING WITH SEVERAL      **
C     **                                                                **
C     **        OBJECTIVE FUNCTIONS",INTERNATIONAL JOURNAL OF FUZZY     **
C     **                                                                **
C     **        SETS AND SYSTEMS",VOL.1,1978,PP.45-55.                  **
C     **                                                                **
C     **                                                                **
C     **        3) ZIMMERMANN,H.J.,"MEDIA SELECTION AND FUZZY LINEAR    **
C     **                                                                **
C     **        PROGRAMMING",THE JOURNAL OF THE OPERATIONAL RESEARCH    **
C     **                                                                **
```

```
C     **        SOCIETY",VOL.29,NO.11,1978,PP.1021-1084.                **
C     **                                                                **
C     **                                                                **
C     **        ADDITIONAL INFORMATION CONCERNING PETERSEN'S            **
C     **                                                                **
C     **        EXCHANGE HEURISTIC ALGORITHM  MAY BE OBTAINED           **
C     **                                                                **
C     **        IN THE FOLLOWING REFERENCE                              **
C     **                                                                **
C     **        PETERSEN,C.C.,"A CAPITAL BUDGETING HEURISTIC            **
C     **                                                                **
C     **        ALGORITHM USING EXCHANGE OPERATIONS",AIIE TRANSACTIONS**
C     **                                                                **
C     **        VOL. 6,NO.2,JUNE 1974,PP.143-150.                       **
C     **                                                                **
C     **                                                                **
C     ********************************************************************
C     ********************************************************************
C
C
C
C
C
C     ********************************************************************
C     **                                                                **
C     **       VARIABLE DEFINITION GUIDE                                **
C     **       *************************                                **
C     **                                                                **
C     **                                                                **
C     **       INPUT VARIABLES                                          **
C     **       ****************                                         **
C     **                                                                **
C     **       NVAR: NUMBER OF VARIABLES                                **
C     **                                                                **
C     **       KTYPE1: NUMBER OF RIGID CONSTRAINTS TYPE1(<)             **
C     **                                                                **
```

```
C     **        X1(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **        ITH TYPE1 RIGID CONSTRAINT.                              **
C     **                                                                 **
C     **        RHS1(I): RIGHT HAND SIDE OF THE ITH TYPE1 CONSTRAINT.    **
C     **                                                                 **
C     **        KTYPE2: NUMBER OF RIGID CONSTRAINTS TYPE2(>)             **
C     **                                                                 **
C     **        X2(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **        ITH TYPE2 RIGID CONSTRAINT.                              **
C     **                                                                 **
C     **        RHS2(I): RIGHT HAND SIDE OF THE ITH TYPE2 CONSTRAINT.    **
C     **                                                                 **
C     **        KTYPE3: NUMBER OF RIGID CONSTRAINTS TYPE3(=)             **
C     **                                                                 **
C     **        X3(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **        ITH TYPE3 RIGID CONSTRAINT.                              **
C     **                                                                 **
C     **        RHS3(I): RIGHT HAND SIDE OF THE ITH TYPE3 CONSTRAINT.    **
C     **                                                                 **
C     **        ITYPE1: NUMBER OF TYPE1 FUZZY OBJECTIVE FUNCTIONS.       **
C     **                                                                 **
C     **        X4(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **        ITH TYPE1 FUZZY OBJECTIVE FUNCTION.                      **
C     **                                                                 **
C     **        ASP1(I): ASPIRATION LEVEL ITH TYPE1 FUZZY OBJECTIVE       **
C     **                                                                 **
C     **        ITYPE2: NUMBER OF TYPE2 FUZZY OBJECTIVE FUNCTIONS.       **
C     **                                                                 **
C     **        X5(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C     **        ITH TYPE2 FUZZY OBJECTIVE FUNCTION.                      **
C     **                                                                 **
C     **        ASP2(I): ASPIRATION LEVEL ITH TYPE2 FUZZY OBJECTIVE       **
C     **                                                                 **
C     **        ITYPE3: NUMBER OF TYPE3 FUZZY OBJECTIVE FUNCTIONS.       **
```

```
C      **      X6(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE          **
C      **      ITH TYPE3 FUZZY OBJECTIVE FUNCTION.                      **
C      **                                                               **
C      **      ASP3(I): ASPIRATION LEVEL ITH TYPE3 FUZZY OBJECTIVE      **
C      **                                                               **
C      **      STINT(I): LOWEST ADMISSABLE VALUE OF THE ITH             **
C      **      TYPE3 FUZZY OBJECTIVE.                                   **
C      **                                                               **
C      **      UTINT(I): GREATEST ADMISSABLE VALUE OF THE ITH           **
C      **      TYPE3 FUZZY OBJECTIVE.                                   **
C      **                                                               **
C      **                                                               **
C      **      INTERNAL VARIABLES                                       **
C      **      ******************                                       **
C      **                                                               **
C      **      ROW(I,J): COEFFICENT OF THE JTH VARIABLE IN THE ITH      **
C      **      CONSTRAINT/OBJECTIVE FORMULATED FOR INPUT INTO           **
C      **      THE EXCHANGE HEURISTIC.                                  **
C      **                                                               **
C      **      RHS(I): RIGHT-HAND SIDE VALUE OF THE ITH CONSTRAINT/     **
C      **      OBJECTIVE FORMULATED FOR INPUT INTO THE                  **
C      **      EXCHANGE HEURISTIC.                                      **
C      **                                                               **
C      **      NPES: TOTAL NUMBER OF CONSTRAINTS/OBJECTIVES TO          **
C      **      BE UTILIZED IN THE EXCHANGE HEURISTIC.                   **
C      **                                                               **
C      **      NFUZZY: TOTAL NUMBER OF FUZZY FUNCTIONS.                 **
C      **                                                               **
C      **      OLJCOF(J): SUM OF THE COEFFICIENTS OF THE JTH VARIABLE   **
C      **      IN THE FUZZY FUNCTION.                                   **
C      **                                                               **
C      **      TECHCO(I,J): COEFFICIENT OF THE JTH VARIABLE IN THE      **
C      **      ITH CONSTRAINT/OBJECTIVE.                                **
C      **                                                               **
C      **      PVALUE(J): SUM OF THE TECHCO(I,J)/AMTRES(I) RATIO        **
C      **      FOR EACH CONSTRAINT/OBJECTIVE.                           **
```

```
C      **      RATIO(J): RATIO OF OBJCOF(J)/RVALUE(J) FOR THE            **
C      **      JTH VARIABLE.                                             **
C      **                                                                **
C      **      AMTRES(I): TOTAL AMOUNT AVAILABLE IN CONSTRAINT I         **
C      **                                                                **
C      **      AMTAVAL(I): AMOUNT AVAILABLE IN CONSTRAINT I FOR THE      **
C      **      EXCHANGE UNDER CONSIDERATION.                             **
C      **                                                                **
C      **      AMTNED(I): AMOUNT NEEDED IN CONSTRAINT I FOR THE          **
C      **      EXCHANGE UNDER CONSIDERATION.                             **
C      **                                                                **
C      **      FTOBFU: VALUE OF THE MEMBERSHIP FUNCTION WITH THE         **
C      **      FITBACK SOLUTION.                                         **
C      **                                                                **
C      **      GNOBFU(L): L=1,CONTAINS THE VALUE OF THE INCREASE         **
C      **                 IN THE MEMBERSHIP FUNCTION OF THE MOST         **
C      **                 ADVANTAGEOUS AND FEASIBLE EXCHANGE.            **
C      **                                                                **
C      **                 L=2, CONTAINS THE VALUE OF THE INCREASE IN     **
C      **                 THE MEMBERSHIP FUNCTION OF THE EXCHANGE        **
C      **                 UNDER CONSIDERATION.                           **
C      **                                                                **
C      **      IMPDSO(K):(K=1,NNSPRO),STORE THE VARIABLES FROM THE SET**
C      **                 OF NON-SELECTED PROJECTS WHICH GAVE THE        **
C      **                 LAST ADVANTAGEOUS AND FEASIBLE EXCHANGE.       **
C      **                                                                **
C      **                (K=1, NSEPRO),STORE THE VARIABLES FROM THE      **
C      **                 SET OF SELECTED PROJECTS WHICH GAVE THE LAST**
C      **                 ADVANTAGEOUS AND FEASIBLE EXCHANGE.            **
C      **                                                                **
C      **      ISELTD(K): STORE ALL VARIABLES BELONGONG TO THE SET       **
C      **      OF SELECTED PROJECTS.(K=1.NVARSE)                         **
C      **                                                                **
```

```
C      **      ISTART: =0,FIRST TIME PROGRAM IS EXECUTING                **
C      **              THE FIRST SEARCH.                                 **
C      **              =1, PROGRAM IS REPEATING THE FIRST SEARCH.        **
C      **                                                                **
C      **      LFITBS(K): STORE ALL VARIABLES THAT ARE PART OF THE       **
C      **      FITBACK SOLUTION.                                         **
C      **                                                                **
C      **      LISTDR(J): STORE THE VARIABLES RANKED IN ORDER OF         **
C      **      DECREASING VALUE OF RATIO(J).                             **
C      **                                                                **
C      **      NNSPRO: NUMBER OF VARIABLES FROM THE SET OF NON-          **
C      **      SELECTED PROJECTS WHICH GAVE THE LAST ADVANTAGEOUS AND    **
C      **      FEASIBLE EXCHANGE.                                        **
C      **                                                                **
C      **      NSEPRO: NUMBER OF VARIABLES FROM THE SET OF SELECTED      **
C      **      PROJECTS WHICH GAVE THE LAST ADVANTAGEOUS AND             **
C      **      FEASIBLE EXCHANGE.                                        **
C      **                                                                **
C      **      NVAFTB: NUMBER OF VARIABLES IN THE FITBACK SOLUTION.      **
C      **                                                                **
C      **      NVARNS: NUMBER OF VARIABLES IN THE SET OF NON-SELECTED    **
C      **      PROJECTS.                                                 **
C      **                                                                **
C      **      NVARSE: NUMBER OF VARIABLES IN THE SET OF SELECTED        **
C      **      PROJECTS.                                                 **
C      **                                                                **
C      **      OBJFUN: THE VALUE OF THE MEMBERSHIP FUNCTION.             **
C      **                                                                **
C      **      SLACK(I): STORE THE SLACK VALUE FOR CONSTRAINT I.         **
C      **                                                                **
C      **      SLCKFT(I): STORE THE SLACK VALUE FOR CONSTRAINT I         **
C      **      DURING THE SEARCH FOR THE FITBACK SOLUTION.               **
C      **                                                                **
C      **                                                                **
C      ******************************************************************
C
```

```
C
C     ******************************************************************
C     **                                                              **
C     **      USER'S GUIDE                                            **
C     **      ************                                            **
C     **                                                              **
C     **      CARD SET                   DATA INPUT FORMAT            **
C     **      ********                   *****************            **
C     **      CARD SET # 1                         9F8.4              **
C     **      NUMBER OF CARDS IN SET: 1                               **
C     **                                                              **
C     **          NVAR                                                **
C     **          KTYPE1                                              **
C     **          KTYPE2                                              **
C     **          KTYPE3                                              **
C     **          ITYPE1                                              **
C     **          ITYPE2                                              **
C     **          ITYPE3                                              **
C     **                                                              **
C     **      CARD SET # 2                         9F8.4              **
C     **      NUMBER OF CARDS IN SET : KTYPE1+1                       **
C     **      RIGID CONSTRAINT TYPE # 1                               **
C     **                                                              **
C     **          ((X1(J,I),I=1,NVAR),J=1,KTYPE1)                     **
C     **          (RHS1(I),I=1,KTYPE1)                                **
C     **                                                              **
C     **      CARD SET # 3                         9F8.4              **
C     **      NUMBER OF CARDS IN SET : KTYPE2+1                       **
C     **      RIGID CONSTRAINT TYPE # 2                               **
C     **          ((X2(J,I),I=1,NVAR),J=1,KTYPE2)                     **
C     **          (RHS2(I),I=1,KTYPE2)                                **
C     **                                                              **
C     **      CARD SET # 4                         9F8.4              **
C     **      NUMBER OF CARDS IN SET : KTYPE3+1                       **
C     **      RIGID CONSTRAINT TYPE # 3                               **
C     **          ((X3(J,I),I=1,NVAR),J=1,KTYPE3)                     **
C     **                                                              **
```

```
C     **         (RHS3(I),I=1,KTYPE3)                                  **
C     **      CARD SET # 5                        9F8.4                **
C     **      NUMBER OF CARDS IN SET : ITYPE1+1                        **
C     **      FUZZY CONSTRAINT TYPE # 1                                **
C     **         ((X4(J,I),I=1,NVAR),J=1,ITYPE1)                       **
C     **         (ASP1(I),TINT1(I),I=1,ITYPE1)                         **
C     **                                                               **
C     **      CARD SET # 6                        9F8.4                **
C     **      NUMBER OF CARDS IN SET : ITYPE2+1                        **
C     **      FUZZY CONSTRAINT TYPE # 2                                **
C     **         ((X5(J,I),I=1,NVAR),J=1,ITYPE2)                       **
C     **         (ASP2(I),TINT2(I),I=1,ITYPE2)                         **
C     **                                                               **
C     **      CARD SET # 7                        9F8.4                **
C     **      NUMBER OF CARDS IN SET : ITYPE3+1                        **
C     **      FUZZY CONSTRAINT TYPE # 3                                **
C     **         ((X6(J,I),I=1,NVAR),J=1,ITYPE3)                       **
C     **         (ASP3(I),STINT(I),UTINT(I),I=1,ITYPE3)                **
C     **                                                               **
C     **                                                               **
C     *****************************************************************
C
C
C
```

```
C
C
C
C     ******************************************
C     **                                      **
C     **    DATA SET 80 IS THE TEMPORARY      **
C     **                                      **
C     **    DATA SET THAT IS USED AS INPUT    **
C     **                                      **
C     **                                      **
C     **    FOR THE EXCHANGE HEURISTIC        **
C     **                                      **
C     ******************************************
C
C
      DIMENSION X1(25,25),X2(25,25),X3(25,25),X4(25,25),
     1X5(25,25),X6(25,25),RHS(25),ROW(25,25),RHS1(25),RHS2(25),
     1RHS3(25),ASP1(25),TINT1(25),ASP2(25),TINT2(25),
     1ASP3(25),STINT(25),UTINT(25)
      READ 1,NVAR,KTYPE1,KTYPE2,KTYPE3,ITYPE1,ITYPE2,ITYPE3
      NFUZZY=ITYPE1+ITYPE2+ITYPE3*2
      NRES=NFUZZY+KTYPE1+KTYPE2+KTYPE3*2


      WRITE(80,103) NVAR,NRES,NFUZZY
  103 FORMAT(3I5)
      M=0
C ***
C ***
C ***RIGID CONSTRAINT TYPE 1(<).
C ***
C ***
      IF(KTYPE1.EQ.0) GO TO 30
      DO 50 IC=1,KTYPE1
   50 READ 2,(Y1(IC,J),J=1,NVAR)
```

```
      READ 2,(RHS1(I),I=1,KTYPE1)
      DO 100 I=1,KTYPE1
      M=M+1
      DO 200 J=1,NVAR
      ROW(M,J)=X1(I,J)
      RHS(M)=RHS1(I)
  200 CONTINUE
  100 CONTINUE
   30 M=KTYPE1
C ***
C ***
C *** RIGID CONSTRAINT TYPE 2 (>)
C ***
C ***
      IF(KTYPE2.EQ.0) GO TO 31
      DO 51 IA=1,KTYPE2
   51 READ 2,(X2(IA,I),I=1,NVAR)
      READ 2,(RHS2(I),I=1,KTYPE2)
      DO 151 IA=1,KTYPE2
      IF(RHS2(IA).GE.0.0) GO TO 151
      RHS2(IA)=-RHS2(IA)
      DO 152 I=1,NVAR
      X2(IA,I)=-X2(IA,I)
  152 CONTINUE
  151 CONTINUE
      DO 300 I=1,KTYPE2
      M=M+1
      DO 400 J=1,NVAR
      ROW(M,J)=X2(I,J)
      RHS(M)=RHS2(I)
  400 CONTINUE
  300 CONTINUE
   31 M=KTYPE1+KTYPE2
C ***
C ***
```

```
C *** RIGID CONSTRAINT TYPE 3 (=).
C ***
C ***
      IF(KTYPE3.EQ.0) GO TO 32
      DO 52 IB=1,KTYPE3
   52 READ 2,(X3(IB,I),I=1,NVAR)
      READ 2,(RHS3(I),I=1,KTYPE3)
      DO 500 I=1,KTYPE3
      M=M+1
      DO 600 J=1,NVAR
      ROW(M,J)=X3(I,J)
      RHS(M)=RHS3(I)
      M=M+1
      ROW(M,J)=-X3(I,J)
      RHS(M)=-RHS3(I)
      IF(J.EQ.NVAR)GO TO 600
      M=M-1
  600 CONTINUE
  500 CONTINUE
   32 M=KTYPE1+KTYPE2+KTYPE3*2
C ***
C ***
C *** FUZZY CONSTRAINT TYPE 1 (<).
C ***
C ***
      IF(ITYPE1.EQ.0) GO TO 33
      DO 60 ID=1,ITYPE1
   60 READ 2,(X4(ID,I),I=1,NVAR)
      READ 2,(ASP1(I),TINT1(I),I=1,ITYPE1)
      DO 700 I=1,ITYPE1
      M=M+1
      DO 800 J=1,NVAR
      ROW(M,J)=-X4(I,J)/TINT1(I)
      RHS(M)=(TINT1(I)-ASP1(I))/TINT1(I)
  800 CONTINUE
```

```
      WRITE(80,104)  (ROW(M,MN),MN=1,NVAR),RHS(M)
  104 FORMAT(8F10.4)
  700 CONTINUE
   33 M=KTYPE1+KTYPE2+ KTYPE3*2+ITYPE1
C ***
C ***
C *** FUZZY CONSTRAINT TYPE 2 (>).
C ***
C ***
      IF(ITYPE2.EQ.0) GO TO 34
      DO 53 IF=1,ITYPE2
   53 READ 2,(X5(IF,I),I=1,NVAR)
      READ 2,(ASP2(I),TINT2(I),I=1,ITYPE2)
      DO 900 I=1,ITYPE2
      M=M+1
      DO 1000 J=1,NVAR
      ROW(M,J)=X5(I,J)/TINT2(I)
      RHS(M)=(TINT2(I)+ASP2(I))/TINT2(I)
 1000 CONTINUE
C     WRITE(6,104)  (ROW(M,MN),MN=1,NVAR),RHS(M)
      WRITE(80,104)  (ROW(M,MN),MN=1,NVAR),RHS(M)
  900 CONTINUE
   34 M=KTYPE1+ KTYPE2+ KTYPE3*2+ ITYPE1+ ITYPE2
C ***
C ***
C *** FUZZY CONSTRAINT TYPE 3 (=).
C ***
C ***
      IF(ITYPE3.EQ.0) GO TO 35
      DO 54 IG=1,ITYPE3
   54 READ 2,(X6(IG,I),I=1,NVAR)
      READ 2,(ASP3(I),STINT(I),UTINT(I),I=1,ITYPE3)
      DO 1100 I=1,ITYPE3
      M=M+1
      DO 1200 J=1,NVAR
      ROW(M,J)=-X6(I,J)/STINT(I)
      RHS(M)=-(ASP3(I)-STINT(I))/STINT(I)
      M=M+1
```

```
      MCOUNT=M
      ROW(M,J)=X6(I,J)/UTINT(I)
      RHS(M)=(UTINT(I)+ASP3(I))/UTINT(I)
      IF(J.EQ.NVAR) GO TO 1200
      M=M-1
 1200 CONTINUE
      KKJ=MCOUNT-1
C     WRITE(6,104)  (ROW(KKJ,MK),MK=1,NVAR),RHS(KKJ)
      WRITE(80,104)  (ROW(KKJ,MK),MK=1,NVAR),RHS(KKJ)
      KKJ=KKJ+1
C     WRITE(6,104)  (ROW(KKJ,MK),MK=1,NVAR),RHS(KKJ)
      WRITE(80,104)  (ROW(KKJ,MK),MK=1,NVAR),RHS(KKJ)
      KKJ=0
 1100 CONTINUE
    1 FORMAT(7I5)
    2 FORMAT(9F8.4)
   35 KJK=KTYPE1+KTYPE2+KTYPE3*2
      DO 1124 LMN=1,KJK
C     WRITE(6,104)  (ROW(LMN,J),J=1,NVAR),RHS(LMN)
 1124 WRITE(80,104)  (ROW(LMN,J),J=1,NVAR),RHS(LMN)
      STOP
      END

TED=     746

OBJECT CODE=    12624 BYTES,ARRAY AREA=    18600 BYTES,TOTAL AREA AVAILABLE=  17817

  NUMBER OF ERRORS=       0, NUMBER OF WARNINGS=       0, NUMBER OF EXTENSIONS=

  0.21 SEC,EXECUTION TIME=    0.06 SEC,
```

```
      COMMON/COM1/ISELTD( 50),NSELTD( 50),ISELEC(3),NSELEC(3),NPRCNS,NP
     +ROSE,IOUTPT
      COMMON/COM2/GNOBFU(2),OBJCOF(50)
      COMMON/COM3/AMTAVL(30),AMTNED(30),SLACK(30),TECHCO(30,50),NRES,IFE
     1ASB
      COMMON/COM4/IM2DSC(6),NNSERO,NSEPRO
      COMMON/COM5/LISTDR( 50),NVARNS,NVARSE
      COMMON/CON21/INDEX,FTCEFU,MCOUNT
      COMMON/CON25/X
      COMMON/CON24/IXX
      COMMON/CON22/FRES
      COMMON/CON23/OBJFUN
      COMMON/CON38/FCONST
      COMMON/CON39/NFUZZY
      COMMON/CON40/IMARK
      COMMON/CON41/KFLAG,TRIAL,JFLAG
      DIMENSION NS2ARY(50),X(50),FRES(50,50),FCONST(50)
      DIMENSION AMTRES(30),IFITBS(50),RATIC(50),RVALUE(50),SLCKFT(30)
      DIMENSION IA(50)
      NVARSE=0
      NVARNS=0
      WRITE(6,840)
C ***
C *** READ DATA
      READ(80,850) NVAR,NRES,NFUZZY
      IOUTPT=0
      IF(NVAR .EQ. 0) STOP
      DO 2 II=1,NRES
      READ(80,860) (TECHCO(II,JJ),JJ=1,NVAR),AMTRES(II)
  2   CONTINUE
      DO 622 I=1,NVAR
      ISELTD(I)=0
      NSELTD(I)=0
 622  CONTINUE
      DO 1134 JJ=1,NVAR
      OBJCOF(JJ)=0.0
```

```
      DO 1133 I=1,NFUZZY
      OBJCOF(JJ)=OBJCOF(JJ)+TECHCO(I,JJ)
 1133 CONTINUE
 1134 CONTINUE
      DO 133 I=1,NFUZZY
      DO 134 JJ=1,NVAR
      FRES(I,JJ)=TECHCO(I,JJ)
      FCONST(I)=AMTRES(I)
  134 CONTINUE
  133 CONTINUE
      ISTART=0
C *** CALCULATE RVALUE FOR EACH VARIABLE
      DO  18 J=1,NVAR
      RVALUE(J)=0.0
      DO 18 I=1,NRES
      IF(AMTRES(I).EQ.0.0) GO TO 18
      RVALUE(J)=RVALUE(J)+TECHCO(I,J)/AMTRES(I)
   18 CONTINUE
C ***
C*** CALCULATE RATIO
      KFLAG=0
      DO 20 J=1,NVAR
      LISTDR(J)=J
      IF(RVALUE(J).EQ.0.0) GO TO 1120
      RATIO(J)=OBJCOF(J)/RVALUE(J)
      GO TO 20
 1120 RATIO(J)=0.0
   20 CONTINUE
C ***
C *** RANK PROJECTS BASED ON VALUE OF RATIO(J). HIGHEST=1ST, ...
      DO 40 ID=2,NVAR
      IID=ID
   30 IF (RATIO(IID).LE.RATIO(IID-1)) GO TO 40
C ***
C *** INTERCHANGE RATIO
      TRATIO=RATIO(IID-1)
```

```
      RATIO(IID-1)=RATIO(IID)
      RATIO(IID)=TRATIO
C ***
C *** INTERCHANGE PROJECT NUMBER
      LISTT=LISTDR(IID-1)
      LISTDR(IID-1)=LISTDR(IID)
      LISTDR(IID)=LISTT
      IID=IID-1
      IF (IID-1) 40,40,30
   40 CONTINUE
C ***
C *** DETERMINE THE INITIAL SOLUTION WITHOUT FITBACK.
C ***
 181  NT=NFUZZY+1
      NCOUNT=0
      SUM=0.0
      NVARNS=0
      DO 111 KJ=1,NVAR
 111  IA(KJ)=0
      NTIM=(2**NVAR)-1
      DO 201 MP=1,NTIM
      INDEX=0
      NVARSE=0
      INDXE=0
      JJ=1
 35   JT=LISTDR(JJ)
      IA(JT)=IA(JT)+1
      IF(IA(JT).NE.2) GO TO 50
      IA(JT)=0
      JJ=JJ+1
      GO TO 35
  50  DO 778 KLJ=1,NVAR
      J=LISTDR(KLJ)
      IF(IA(J).EQ.0) GO TO 165
      NVARSE=NVARSE+1
      ISELID(NVARSE)=LISTDR(KLJ)
```

```
      GO TO 778
  165 INDEX=INDEX+1
      NSELTD(INDEX)=LISTDR(KLJ)
 778  CONTINUE
      DO 80 I=NT,NRES
      SUM=0.0
      DO 60 IND=1,NVAR
      J=LISTDR(IND)
      IF(IA(J).EQ.0) GO TO 6C
      SUM=SUM+TECHCO(I,J)
      GO TO 60
  60  CONTINUE
      IF(SUM.LE.AMTRES(I)) GO TO 70
      NCOUNT=0
      GO TO 201
 70   NSLACK=0
      DO 1110 IJK=NT,NRES
      AMTSUD=0.0
      IF(NVARSE.EQ.0) GO TO 201
      DO 1100 IM=1,NVARSE
      KND=ISELTD(IM)
 1100 AMTSUD=AMTSUD+TECHCO(IJK,KND)
      SLACK(IJK)=AMTRES(IJK)-AMTSUD
      IF(SLACK(IJK).GE.0.0)GC TC 1121
      NCOUNT=0
      GO TO 201
 1121 NSLACK=NSLACK+1
 1110 CONTINUE
      IF(NSLACK.EQ.(NRES-NFUZZY)) GO TO 85
      GO TO 80
  85  DO 1198 JT=1,NFUZZY
      AMTSUD=0.0
      DO 1199 IQ=1,NVARSE
      IT=ISELTD(IQ)
 1199 AMTSUD=AMTSUD+TECHCO(JT,IT)
      SLACK(JT)=AMTRES(JT)-AMTSUD
```

```
      IF(SLACK(JT).GE.0.0) GO TO 1198
      NCOUNT=0
      GO TO 201
 1198 CONTINUE
   80 CONTINUE
      IF(NVARSE.EQ.0) GO TO 201
      GO TO 1185
  201 CONTINUE
C *** FILL THE VECTORS OF SELECTED AND NON-SELECTED VARIABLES
 1185 WRITE(6,910) (ISELTD(II),II=1,NVARSE)
      NVARNS=INDEX
      WRITE(6,924) (NSELTD(IJ),IJ=1,NVARNS)
 924  FORMAT(1X,'THE SET OF NON-SELECTED VARIABLES IS ',16I5)
C ***
C *** CALCULATE THE MEMBERSHIP FUNCTION AS AN INITIAL SOLUTION
      OBJFUN=0.0
      CALL OBJ
      IF(IMARK.EQ.10) GO TO 95
      WRITE(6,920) OBJFUN
C *** CALCULATE SLACK FOR EACH CONSTRAINT
   95 DO 110 I=1,NRES
      AMTSUD=0.
      DO 100 IS=1,NVARSE
      IND=ISELTD(IS)
  100 AMTSUD=AMTSUD+TECHCC(I,IND)
      SLACK(I)=AMTRES(I)-AMTSUD
      IF (IOUIPT.EQ.0) GO TO 110
      IF(IMARK.EQ.10) GO TO 110
      WRITE(6,925) I,SLACK(I)
 110  SLCKFT(I)=SLACK(I)
C ***
C *** RANK PROJECTS IN BOTH SETS OF SELECTED AND NON - SELECTED
C *** PROJECTS BASED ON THEIR OBJECTIVES' COEFFICIENTS VALUE
  195 CALL RANK
```

```
      IF (ISTART.EQ.1) GO TO 150
C ***
C *** FIND FITBACK SOLUTION
      DO 220 IP=1,NVARSE
  220 LFTTBS(IP)=ISELTD(IP)
      NVAFTB=NVARSE
      FTOBFU=OBJFUN
      MCOUNT=0
      DO 140 J=1,NVARNS
      IFLAG=0
      INDEX=NSELTD(J)
      DO 130 II=1,NRES
      IF(SLCKFT(II).LT.TECHCC(II,INDEX)) GO TO 130
      IFLAG=IFLAG+1
  130 CONTINUE
      IF(IFLAG.NE.NRES) GO TO 140
C *** FOUND A VARIABLE FOR FITBACK SOLUTION
C *** UPDATE SLACK VALUE AND CALCULATE
C *** THE MEMBERSHIP FUNCTION
      DO 135 IK=1,NRES
  135 SLCKFT(IK)=SLCKFT(IK)-TECHCC(IK,INDEX)
      NVAFTB=NVAFTB+1
      LFITBS(NVAFTB)=NSELTD(J)
      CALL FTOBJ
      MCOUNT=MCOUNT+1
  140 CONTINUE
      IF(NVAFTB.EQ.NVARSE) GO TO 141
      WRITE(6,930) (LFITBS(II),II=1,NVAFTB)
      WRITE(6,920) FTOBFU
      WRITE(6,932)
      GO TO 150
  141 WRITE(6,142)
  142 FORMAT(5X,'NO FITBACK SOLUTION CAN BE FOUND TO IMPROVE THE INITIAL
     1 SOLUTION')
      FTOBFU=-9999.
C ***
```

```
C *** 2/1 EXCHANGE
 150  NPRONS=2
      NPROSE=1
      NPROFT=0
      GNOBFU(1)=0.
      NS1LST=NVARNS-1
      IF (NS1LST.EQ.0) GO TO 212
      DO 152 INS=1,NS1LST
 152  NS2ARY(INS)=NVARNS
      DO 210 JS1=1,NVARSE
      ISELEC(1)=ISELTD(JS1)
      DO 190 NS1=1,NS1LST
      NS1NP=NS1
      NSELEC(1)=NSELTD(NS1)
      NS2LST=NS2ARY(NS1)
      NS2ST=NS1+1
      IF (NS2ST.GT.NS2LST) GO TO 210
      IFLAG=0
      DO 170 NS2=NS2ST,NS2LST
      NS2NP=NS2
      NSELEC(2)=NSELTD(NS2)
C *** CHECK MEMBERSHIP FUNCTION
      CALL ACHVMT
      IF(GNOBFU(2).GE.GNOBFU(1)) GO TO 160
C *** 2/1 EXCHANGE NOT ADVANTAGEOUS
C *** STORE PROJECT NUMBERS SO THAT THEY WILL NOT BE USED IN THE 1/1 EXC
     . IF(NS1.NE.1.OR.NPROFT.NE.0) GO TO 155
      NPROFT=JS1
 155  IFLAG=1
      IF((NS2-NS1)-2) 200,200,180
C *** CHECK FEASIBILITY
 160  CALL FEASBL
      IF(IFEASB.NE.NRFS) GO TO 170
      CALL IMPVNT
      IFLAG=1
      IF((NS2-NS1)-2) 200,200,180
```

```
  170  CONTINUE
  180  NS2LST=NS2NP-1*IFLAG
       IEND=NVARNS-1
       DO 185 I=NS1,IEND
       IF(NS2ARY(1).NE.NS2LST) GO TO 185
       NS2ARY(I)=NS2LST
   185 CONTINUE
   190 CONTINUE
   200 IDIF=(NS2NP-NS1NP)*IFLAG
       NS1LST=NS1NP-((2-IDIF)*IFLAG)
       IF(NS1LST.EQ.0) GO TO 212
       IF(NS1NP.NE.NS1LST) GO TO 210
       NS2ARY(NS1NP)=NS2NP+((1-IDIF)*IFLAG)
   210 CONTINUE
C ***
C ***  1/1 EXCHANGE
  212  NPRONS=1
       JS1LST=NVARSE+NPROFT
       IF(JS1LST.GT.NVARSF)JS1LST=NPROFT-1
       DO 240 NS1=1,NVARNS
       NSELEC(1)=NSELTD(NS1)
       IFLAG=0
       DO 221 JS1=1,JS1LST
       ISELEC(1)=ISELTD(JS1)
       JS1NP=JS1
       CALL ACHVMT
       IF(GNOBFU(2).GE.GNOBFU(1)) GO TO 215
       IFLAG=1
       GO TO 230
  215  CALL FEASBL
       IF(IFEASB.NE.NRES) GO TO 221
       CALL IMPVNT
       IFLAG=1
       GO TO 230
   221 CONTINUE
C ***
```

```
C *** 1/2 EXCHANGE
  230 JS1LST=JSINP-1*IFLAG
      IF (JS1LST.EQ.0) GO TO 250
  240 CONTINUE
  250 IF(GNOBFU(1).LE.0.) GO TO 290
      OBJFUN=OBJFUN+GNOBFU(1)
C *** FOUND AN EXCHANGE TO IMPROVE THE MEMBERSHIP FUNCTION
      IF(IOUTPT.EQ.0) GO TO 260
      WRITE(6,935)
      CALL OBJ
      WRITE(6,920) OBJFUN
      WRITE(6,940)
C ***
C *** MAKE THE EXCHANGE
  260 CALL EXCHGE
C ***
C *** REPEAT FIRST SEARCH
      ISTART=1
      GO TO 95
  290 IF (ISTART.EQ.0) GO TO 300
      WRITE(6,935)
      WRITE(6,950) (ISFLTD(II),II=1,NVARSE)
      CALL OBJ
      WRITE(6,920) OBJFUN
      GO TO 305
  300 WRITE(6,955)
  305 IF(NVARNS.LT.3) GO TO 480
C *** SECOND SEARCH (3/1,3/2,3/3 EXCHANGES)
      WRITE(6,960)
      NPRONS=3
      GNOBFU(1)=0.
      NS1LST=NVARNS-2
      ISTP32=0
      ISTP33=0
      DO 430 NS1=1,NS1LST
C ***
```

```
C *** 3/1 EXCHANGE
      DO 310 IIP=1,3
  310 NSELEC(IIP)=NSELTD(NS1+IIP-1)
      NPROSF=1
      IFLAG=1
      DO 330 JS1=1,NVARSE
      JS1NP=JS1
      ISELEC(1)=ISELTD(JS1)
C ***
C *** CHECK FOR PROFITABILITY
      CALL ACHVMT
      IF(GNOBFU(2).GE.GNOBFU(1)) GO TO 320
      IFLAG=0
      GO TO 340
C ***
C *** CHECK FOR FEASIBILITY
  320 CALL FEASBL
      IF(IFEASB.NE.NRES) GO TO 330
C ***
C**** STORE IMPROVED SOLUTION TEMPORARILY
      CALL IMPVNT
  330 CONTINUE
  340 IF(JS1NP.LE.(2-IFLAG).OR.ISTP32.EQ.1) GO TO 430
C ***
C *** 3/2 EXCHANGE
      JS1LST=JS1NP-2+IFLAG
      JS2LST=JS1NP-1+IFLAG
      JSLIMT=JS1NP
      NPROSE=2
      DO 370 JS1=1,JS1LST
      ISELEC(1)=ISELTD(JS1)
      JS2ST=JS1+1
      IF(JS2ST.GT.JS2LST) GO TO 380
      IFLAG=0
      DO 360 JS2=JS2ST,JS2LST
      JS2NP=JS2
```

```
      ISELEC(2)=ISELTD(JS2)
C ***
C *** CHECK FOR PROFITIBILITY
      CALL ACHVMT
      IF(GNOBFU(2).GE.GNOBFU(1)) GO TO 350
      IF(JS1.NE.1.OR.JS2.NE.2) GO TO 345
      ISTP32=1
      GO TO 430
  345 IF((JS2-JS1).LE.2) GO TO 380
      IFLAG=1
      GO TO 365
C ***
C *** CHECK FOR FEASIBILITY
  350 CALL FEASBL
      IF(IFEASB.NE.NRES) GO TO 360
C ***
C *** STORE IMPROVED SOLUTION TEMPORARLY
      CALL IMPVNT
  360 CONTINUE
  365 JS2LST=JS2NP-1*IFLAG
  370 CONTINUE
  380 IF(NVARSE.LT.3.OR.JS1LST.EQ.1.OR.ISTP33.EQ.1) GO TO 430
C ***
C *** 3/3 EXCHANGE
      NPROSE=3
      JS1LST=JS1LST-1
      JS2LST=JS2LST-1
      JS3LST=JSLIMT
      DO 420 JS1=1,JS1LST
      ISELEC(1)=ISELTD(JS1)
      JS2ST=JS1+1
      DO 410 JS2=JS2ST,JS2LST
      ISELEC(2)=ISELTD(JS2)
      JS3ST=JS2+1
      DO 400 JS3=JS3ST,JS3LST
      ISELEC(3)=ISELTD(JS3)
C ***
```

```
C *** CHECK FOR PROFITABILITY
      CALL ACHVMT
      IF(GNOBFU(2).GE.GNOBFU(1)) GO TO 390
      IF(JS1.NE.1.OR.JS2.NE.2.OR.JS3.NE.3) GO TO 385
      ISTP33=1
      GO TO 430
  385 IF((JS3-JS2)-2) 430,430,410
C ***
C *** CHECK FOR FEASIBILITY
  390 CALL FEASBL
      IF(IFEASB.NE.NRES) GO TO 400
C ***
C *** STORE IMPROVED SOLUTION TEMPORARLY
      CALL IMPVMT
  400 CONTINUE
  410 CONTINUE
  420 CONTINUE
  430 CONTINUE
      IF(GNOBFU(1).GT.0.) GO TO 470
      WRITE(6,970)
      GO TO 480
  470 WRITE(6,980)
      CALL OBJ
  480 IF(OBJFUN.GE.FTOBFU) GO TO 490
      WRITE(6,990) (LFITBS(II),II=1,NVAFTB)
      WRITE(6,920)FTOBFU
      GO TO 500
  490 WRITE(6,990)(ISELTD(II),II=1,NVALSE)
      WRITE(6,920) OBJFUN
  500 STOP
  840 FORMAT(1H0,36X,59H*** HEURISTIC ALGORITHM FOR LARGE 0 - 1 LINEAR P
     +ROGRAMS ***//)
  850 FORMAT(16I5)
  860 FORMAT(8F10.4)
  910 FORMAT(1H0,64HTHE INITIAL SOLUTION (WITHOUT FITBACK) IS COMPOSED O
     +F PROJECTS :,50I4///)
```

```
920  FORMAT(1H0,41HTHE VALUE OF THE MEMBERSHIP FUNCTION IS :,F8.2///)
930  FORMAT(1H0,46HTHE FITBACK SOLUTION IS COMPOSED OF PROJECTS :,50I4)
932  FORMAT(1H0,45X,41H*** FIRST SEARCH (2/1 - 1/1 EXCHANGE) ***//)
935  FORMAT(1H0,73HTHE FIRST SEARCH HAS FOUND AN EXCHANGE TO IMPROVE TH
    +E MEMBERSHIP FUNCTION)
940  FORMAT(1H0,46X,40HREPEAT FIRST SEARCH (2/1 - 1/1 EXCHANGE)///)
950  FORMAT(1H0,38HTHE SOLUTION IS COMPOSED OF PROJECTS :,50I4)
955  FORMAT(1H0,73HTHE FIRST SEARCH HAS FOUND NO EXCHANGE TO IMPROVE TH
    +E MEMBERSHIP FUNCTION)
990  FORMAT(1H1,44HTHE FINAL SOLUTION IS COMPOSED OF PROJECTS :,50I4)
925  FORMAT(1H0,26HTHE SLACK FOR CONSTRAINT :,I4,5H IS :,F8.2)
960  FORMAT(1H0,42X,'  **** SECOND SEARCH (3/1 3/2 3/3 EXCHANGE) ***
    +')
970  FORMAT(1H0,76HTHE SECOND SEARCH HAS FOUND NO EXCHANGES TO IMPROVE
    +THE MEMBERSHIP FUNCTION.)
980  FORMAT(1H0,75HTHE SECOND SEARCH HAS FOUND AN EXCHANGE TO IMPROVE T
    +HE MEMBERSHIP FUNCTION.)
     END



     SUBROUTINE RANK
     DIMENSION X(50)
     COMMON/COM1/ISELTD(50),NSELTD(50),ISELEC(3),NSELEC(3),NPRONS,NPR
    +OSE,IOUTPT
     COMMON/COM2/GNOBFN(2),OBJCOF(50)
     COMMON/COM5/IISTDR( 50),NVARNS,NVARSE
     COMMON/CON25/X
C ***
C *** RANK PROJECTS IN SET OF NON - SELECTED PROJECTS (NSELTD) BASED
C *** ON OBJCO VALUE
     IF(NVARNS .EQ. 1) GO TO 21
     DO 20 ID=2,NVARNS
     IID=ID
  10 IND1=NSELTD(IID-1)
```

```
      IND2=NSELTD(IID)
      IF(OBJCOF(IND2).LE.OBJCOF(IND1)) GO TO 20
      NSELTT=NSELTD(IID-1)
      NSELTD(IID-1)=NSELTD(IID)
      NSELTD(IID)=NSELTT
      IID=IID-1
      IF (IID-1) 20,20,10
   20 CONTINUE
21    CONTINUE
C ***
C *** RANK PROJECTS IN SET OF SELECTED PROJECTS (ISELTD) BASED ON
C *** OBJECO VALUE
      IF(NVARSE.EQ.1) GO TO 41
      DO 40 ID=2,NVARSE
      IID=ID
   30 IND1=ISELTD(IID-1)
      IND2=ISELTD(IID)
      IF(OBJCOF(IND2).GE.OBJCOF(IND1)) GO TO 40
      ISELTT=ISELTD(IID-1)
      ISELTD(IID-1)=ISELTD(IID)
      ISELTD(IID)=ISELTT
      IID=IID-1
      IF (IID-1) 40,40,30
   40 CONTINUE
  41  CONTINUE
      IF (IOUTPT.EQ.0) GO TO 70
      WRITE(6,50) (NSELTD(II),II=1,NVARNS)
      WRITE(6,60) (ISELTD(II),II=1,NVARSE)
   50 FORMAT(1H0,60HTHE SET OF NON - SELECTED PROJECTS IS COMPOSED OF PR
     +OJECTS :,50I4)
   60 FORMAT(1H0,54HTHE SET OF SELECTED PROJECTS IS COMPOSED OF PROJECTS
     + :,50I4///)
   70 RETURN
      END
```

```
      SUBROUTINE IMPVNT
      COMMON/COM1/ISELTD(50),NSELTD(50),ISELEC(3),NSELEC(3),NPRONS,NPR
     +OSE,IOUTPT
      COMMON/COM2/GNOBFU(2),OBJCOF(50)
      COMMON/COM4/IMPDSO(6),NNSPRO,NSEPRO
      GNOBFU(1)=GNOBFU(2)
      DO 10 IP=1,NPRONS
   10 IMPDSO(IP)=NSELEC(IP)
      DO 20 IP=1,NPROSE
   20 IMPDSO(NPRONS+IP)=ISELEC(IP)
      NNSPRO=NPRONS
      NSEPRO=NPROSE
      IF (IOUTPT.EQ.0) GO TO 40
      WRITE(6,30)
   30 FORMAT(1H0,48H*** THE EXCHANGE IS BOTH PROFITABLE AND FEASIBLE)
   40 RETURN
      END
```

```
      SUBROUTINE FEASBL
      COMMON/COM1/ISELTD(50),NSELTD(50),ISELEC(3),NSELEC(3),NPRONS,NPR
     +OSE,IOUTPT
      COMMON/COM3/AMTAVL(30),AMTNED(30),SLACK(30),TECHCO(30,50),NRES,IFE
     +ASB
      IFEASB=0
      DO 40 I=1,NRES
      AMTNED(I)=0.
      DO 10 IP=1,NPRONS
      INDX=NSELEC(IP)
   10 AMTNED(I)=AMTNED(I)+TECHCO(I,INDX)
      DO 22 IP=1,NPROSE
      INDX=ISELEC(IP)
   22 AMTAVL(I)=SLACK(I)+TECHCO(I,INDX)
      IF(I.EQ.NRES.AND.AMTNED(I).GT.AMTAVL(I)) GO TO 40
      IF(IOUTPT.EQ.0) GO TO 70
   70 IF(AMTNED(I).GT.AMTAVL(I)) GO TO 40
      IFEASB=IFEASB+1
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE EXCHGE
      COMMON/COM1/ISELTD(50),NSELTD(50),ISELEC(3),NSELEC(3),NPRONS,NPR
     +OSE,IOUTPT
      COMMON/COM4/IMPDSO(6),NNSPRO,NSEPRO
      COMMON/COM5/LISTDR( 50),NVARNS,NVARSE
C ***
C *** MAKE THE EXCHANGE IN SET OF NON - SELECTED PROJECTS
      ISST=1
      INDEX=0
      DO 30 IP=1,NVARNS
      IF (ISST.GT.NNSPRO) ISST=NNSPRO
      DO 10 IS=ISST,NNSPRO
      IF (NSELTD(IP).EQ.IMPDSC(IS)) GO TO 20
      INDEX=INDEX+1
      LISTDR(INDEX)=NSELTD(II)
      GO TO 30
   10 CONTINUE
   20 ISST=IS+1
   30 CONTINUE
      IF (INDEX.EQ.0) GO TO 45
      DO 40 IP=1,INDEX
   40 NSELTD(IP)=LISTDR(IP)
   45 DO 50 IP=1,NSEPRO
   50 NSELTD(INDEX+IP)=IMPDSC(NNSPRO+IP)
      NVARNS=NVARNS+NSEPRO-NASPRO
C ***
C *** MAKE THE EXCHANGE IN SET OF SELECTED PROJECTS
      ISST=1
      INDEX=0
      DO 80 IP=1,NVARSE
      IF (ISST.GT.NSEPRO) ISST=NSEPRO
      DO 60 IS=ISST,NSEPRO
      IF (ISELTD(IP).EQ.IMPDSC(NNSPRO+IS)) GO TO 70
      INDEX=INDEX+1
      LISTDR(INDEX)=ISELTD(II)
      GO TO 80
```

```
 60 CONTINUE
 70 ISST=IS+1
 80 CONTINUE
    IF (INDIX.EQ.0) GO TO 95
    DO 90 IP=1,INDEX
 90 ISELTD(IP)=LISTDR(IP)
 95 DO 100 IP=1,NNSPRO
100 ISELTD(INDEX+IP)=IMPDSO(IP)
    NVARSE=NVARSE+NNSPRO-NSEPRO
    RETURN
    END


    SUBROUTINE ACHVMT
    DIMENSION FRES(50,50),FCONST(50),FOBJ(50),TEST(50),IBASE(50,50),
   1NBASE(50,50)
    COMMON/COM1/ISELTD(50),NSELTD(50),ISELEC(3),NSELEC(3),NPRONS,NPR
   +OSE,IOUTPT
    COMMON/COM2/GNOBFU(2),OBJCOF(50)
    COMMON/COM5/LISTDR(50),NVARNS,NVARSE
    COMMON/CON24/IXX
    COMMON/CON22/FRES
    COMMON/CON34/NCOUNT
    COMMON/CON38/FCONST
    COMMON/CON39/NFUZZY
    COMMON/CON41/KFLAG,TRIAL,JFLAG
    COMMON/CON40/IMARK
    COMMON/CON23/OBJFUN
    GNOBFU(2)=0.0
    KFLAG=1
    DO 134 I=1,NFUZZY
    FOBJ(I)=0.0
    DO 145 IP=1,NVARSE
    INDX=ISELTD(IP)
    FOBJ(I)=(-FRES(I,INDX))+FOBJ(I)
```

```
145   CONTINUE
      FOBJ(I)=FCONST(1)+FOBJ(I)
      DO 135 IP=1,NPRONS
      INDX=NSELEC(IP)
      NBASE(I,IP)=NSELEC(IP)
 135  FOBJ(I)=FOBJ(I)+(-FRFS(I,INDX))
      DO 136 IIP=1,NPROSE
      INDX=ISELEC(IIP)
      IBASE(I,IIP)=ISELEC(IIP)
136   FOBJ(I)=FOBJ(I)-(-FRES(I,INDX))
      TEST(I)=FOBJ(I)
134   CONTINUE
      DO 83 I=2,NFUZZY
      J=I-1
      IF(TEST(I).LT.0.0) GO TO 83
      IF(TEST(I).GT.1.0) TEST(I)=1.0
      IF(TEST(I).LE.TEST(J)) GO TO 183
      BSOFAR=TEST(J)
      DO 1283 IP=1,NPROSE
1283  ISELEC(IP)=IBASE(J,IP)
      DO 1383 IP=1,NPRONS
1383  NSELEC(IP)=NBASE(J,IP)
      GO TO 83
 183  BSOFAR=TEST(I)
      DO 283 IP=1,NPROSE
 283  ISELEC(IP)=IBASE(I,IP)
      DO 383 IP=1,NPRONS
 383  NSELEC(IP)=NBASE(I,IP)
  83  CONTINUE
 34   GNOBFU(2)=BSOFAR-OBJFUN
      IF(GNOBFU(2).LE.0.0) GNOBFU(2)=0.0
 35   IF(IOUTPT.EQ.0) GO TO 60
      WRITE(6,30) (ISELEC(II),II=1,NPROSE)
 30   FORMAT(1H0,65H THE EXCHANGE UNDER CONSIDERATION CONSISTS OF REPLAC
     +NG PROJECTS :,3I4)
      WRITE(6,40) (NSELEC(II),II=1,NPRONS)
  40  FORMAT(1H+,79X,14H BY PROJECTS :,3I4)
 51   WRITE(6,50) GNOBFU(2)
```

```
 50 FORMAT(9X,'  THE GAIN IN THE MEMBERSHIP FUNCTION WOULD BE :',F8.2)
 80  RETURN
     END


     SUBROUTINE OBJ
     DIMENSION FRES(50,50),FCONST(50),FOBJ(50),YLAMDA(50)
     COMMON/COM1/ISELTD(50),NSELTD(50),ISELEC(3),NSELEC(3),NPRONS,
    1NPROSE,IOUTPT
     COMMON/CON22/FRES
     COMMON/CON23/OBJFUN
     COMMON/COM5/LISTDR(50),NVARNS,NVARSE
     COMMON/CON38/FCONST
     COMMON/CON39/NFUZZY
     COMMON/CON40/IMARK
     COMMON/CON41/KFLAG,TRIAL,JFLAG
     DO 134 I=1,NFUZZY
     FOBJ(I)=0.0
     DO 135 IP=1,NVARSE
     INDX=ISELTD(IP)
135  FOBJ(I)=(-FRES(I,INDX))+FOBJ(I)
     YLAMDA(I)=FCONST(I)+FOBJ(I)
     IF(YLAMDA(I).LT.0.0) GO TO 139
     IF(YLAMDA(I).GT.1.0) YLAMDA(I) =1.0
     TEST=YLAMDA(I)
     IF(I.EQ.1) GO TO 138
     IF(TEST.LT.BEST) BEST=YLAMDA(I)
     GO TO 134
 138 BEST=TEST
 134 CONTINUE
     OBJFUN=BEST
     IMARK=0
     GO TO 152
 139 WRITE(6,140)
 140 FORMAT(5X,'THE INITIAL SOLUTION IS INFEASIBLE')
     IMARK=10
```

```
      OBJFUN=0.0
152   RETURN
      END

      SUBROUTINE FTCBJ
      DIMENSION FRES(50,50),FCONST(50),OBJ(50),LOBJ(50)
      COMMON/CON21/INDEX,FTCEFU,MCOUNT
      COMMON/CON22/FRES
      COMMON/CON34/NCOUNT
      COMMON/CON2/GNOBFU(2),OBJCOF(50)
      COMMON/CON38/FCONST
      COMMON/CON39/NFUZZY
      IF(MCOUNT.GT.0) GC TO 932
      DO 934 IF=1,NFUZZY
934   OBJ(IF)=FCONST(IF)+FTCEFU
      DO 935 IG=1,NFUZZY
      LOBJ(IG)=OBJ(IG)+FRES(IG,INDEX)
      TEST=LOBJ(IG)
      IF(IG.EQ.1) GC TO 938
      IF(TEST.LT.BEST) BEST=LOBJ(IG)
      GO TO 935
 938  BEST=TEST
 935  CONTINUE
      FTOBFU=BEST
      RETURN
 932  DO 1034 JF=1,NFUZZY
1034  OBJ(JF)=FTOBFU
      DO 1035 JG=1,NFUZZY
      LOBJ(JG)=OBJ(JG)+FRES(JG,INDEX)
      TEST=LOBJ(JG)
      IF(JG.EQ.1) GC TO 1038
      IF(TEST.LT.BEST) BEST=LOBJ(JG)
      GO TO 1035
1038  BEST=TEST
1035  CONTINUE
```

```
      FTOBFU=BEST
      RETURN
      END

 D=   104717

OBJECT CODE=   31024 BYTES,ARRAY AREA=   40164 BYTES,TOTAL AREA AVAILABLE

  NUMBER OF ERRORS=       0, NUMBER OF WARNINGS=       0, NUMBER OF EXT

  0.42 SEC,EXECUTION TIME=     0.84 SEC,
```

```
*** HEURISTIC ALGORITHM FOR LARGE 0 - 1 LINEAR PROGRAMS ***

THE INITIAL SOLUTION (WITHOUT FITBACK) IS COMPOSED OF PROJECTS:    6    2    8
THE SET OF NON-SELECTED VARIABLES IS    7    4    5    1    3   10    9

THE VALUE OF THE MEMBERSHIP FUNCTION IS:    0.13

     NO FITBACK SOLUTION CAN BE FOUND TO IMPROVE THE INITIAL SOLUTION

THE FIRST SEARCH HAS FOUND NO EXCHANGE TO IMPROVE THE MEMBERSHIP FUNCTION


          **** SECOND SEARCH (3/1 3/2 3/3 EXCHANGE) ****

THE SECOND SEARCH HAS FOUND NO EXCHANGES TO IMPROVE THE MEMBERSHIP FUNCTION


      THE FINAL SOLUTION IS COMPOSED OF PROJECTS:    8    6    2

      THE VALUE OF THE MEMBERSHIP FUNCTION IS:    0.13
```

APPENDIX D

Sample Data Input

Sample Data Input

The required data input for Example Problem Number 2 is presented. The data input requirements are identical for the computer codes presented in Appendices B and C.

```
  10      0    1     5    0      1      2
-.0304   -.0618   -0.10   -.1165   -.1054   -.0618   -.0408   -.0202   -.0304
-.01
-.1625
   1.       1.       1.      0.       0.       0.       0.       0.       0.
   0.
   0.       0.       0.      1.       1.       1.       1.       0.       0.
   0.
   0.       0.       0.      0.       0.       0.       0.       1.       1.
   1.
   0.       1.       0.      0.       1.       0.       0.       0.       0.
   0.
   0.       0.       0.      0.       0.       0.       1.       0.       0.
  -1.
   1.       1.       1.      1.       0.
 120.      95.     160.     64.      67.      96.     119.      42.      36.
  70.
 195.      65.
  32.      57.      19.     95.     107.      61.      48.      23.      10.
  15.
  21.      35.      10.     60.      83.      27.      50.      12.       7.
  16.
 150.      30.      15.    100.      30.      10.
```